Agilent

VWorks Automation Control

Version 14

User Guide

Original Instructions

Notices

Manual Part Number

D0004378 Revision B.2

July 2024

Copyright

© Agilent Technologies, Inc. 2024

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Agilent Technologies, Inc. as governed by United States and international copyright laws.

Contact Information

Agilent Technologies Inc.
Automation Solutions
5301 Stevens Creek Blvd.
Santa Clara, CA 95051
USA

Web:
https://www.agilent.com

Contact page:
https://www.agilent.com/en/contact-us/page

Documentation feedback:
documentation.automation@agilent.com

Acknowledgements

Microsoft® and Windows® are either registered trademarks or trademarks of the Microsoft Corporation in the United States and other countries.

Warranty

The material contained in this document is provided "as is," and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Safety Notices

WARNING

A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

CAUTION

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

Contents

iii. Preface ix
About this guide x
Accessing Agilent Automation Solutions user guides xiii

1. Software overview 1
Architecture and features overview 2
Relationship of components in VWorks software 5
Supported devices 9
Overview of VWorks user interface 11

2. Creating a protocol: basic procedure 17
About protocols, processes, and tasks 18
Workflow for creating a basic protocol 22
Preparing for protocol writing 24
Adding devices 26
Setting up location devices 32
Plate Pad, Standard 32
Waste Bin 34
Setting up phantom devices 36
Creating a new protocol 41
Setting protocol options 42
Importing and exporting miscellaneous files 49
Configuring labware 53
Adding processes 58
Setting plate parameters 59
Adding and deleting tasks 67
Specifying time constraints between dependent tasks 71
Setting up Startup and Cleanup Protocol processes 75
Saving the protocol 76
Compiling the protocol 77
Simulating the protocol run 80
Printing protocols 86

3. Creating a protocol: advanced topics 89
Tracking barcodes 90
Using simple variables 93
Using JavaScript 100
VWorks-defined functions 103
Experiments database objects 111
InventoryLabware object 118
plate object 120

plates[ ] object . . . 123
task object . . . 126
Using JavaScript with the CentrifugeAuto task . . . 136
About scripting the Print and Apply task . . . 137
plateDB object . . . 140
runset object . . . 148
forms object . . . 151
Using start and finish protocol scripts . . . 152
Using JavaScript utilities . . . 153
Creating new protocol tasks using the JavaScript Wrapper . . . 158

4. Creating forms for protocol operators . . . 177
About forms for running protocols . . . 178
Workflow for creating or editing a form . . . 181
About the naming of form controls . . . 184
Creating or editing a form . . . 185
Overview of the specialized buttons . . . 188
Configuring a run button and other specialized buttons . . . 194
Adding indicators for elapsed time and progress to a form . . . 202
Adding form controls that allow editing or runtime data display . . . 204
Example: Creating a scripted Pushbutton control in a form . . . 232
Setting the form properties . . . 240
Understanding JavaScript context in form design . . . 243
Opening a message from a form button using JavaScript . . . 247
Configuring a form for experiment IDs and Experiments database . . . 249
Adding specialized controls for the Experiments Editor . . . 253
Configuring a protocol to track experiment data . . . 260

5. Using macros to create protocols . . . 265
About protocol macros and the macro library . . . 266
Adding macros to the macro library . . . 269
Inserting macros in protocols . . . 272
Editing a macro . . . 276
Adding parameter variables and scripting to a macro . . . 282
Managing macro libraries . . . 285

6. Running a protocol . . . 287
Workflow for running a protocol . . . 288
Opening a form . . . 289
Opening a protocol . . . 291
Disabling and enabling a device in the device file . . . 293
Viewing and setting file directory paths . . . 299
Setting general and view options . . . 301
Setting error-handling options . . . 307

Setting up email notification . . . 309
Starting the protocol run . . . 312
Managing runsets . . . 317
Monitoring the overall run progress . . . 329
Tracking the run progress of instances or devices . . . 330
Pausing the run . . . 337
Stopping the run . . . 341

7. Setting parameters for microplate-handling tasks . . . 343
Centrifuge Process . . . 344
CentrifugeAuto . . . 353
Configure Static Labware . . . 361
Delid . . . 363
Dismount . . . 367
Incubate . . . 370
Mount . . . 374
Move to Location (Agilent Bravo) . . . 376
Place Plate . . . 378
Print and Apply . . . 381
Relid . . . 397
Reserve Location . . . 401
Rotate Stage (Agilent Microplate Labeler) . . . 403
Seal (Agilent PlateLoc) . . . 405
Waste . . . 408

8. Setting parameters for microplate storage tasks . . . 411
clearInventory . . . 412
Downstack . . . 414
exportDatabase . . . 418
importCsvToInventory . . . 420
Load . . . 422
Reorder . . . 427
Scan Stack . . . 432
Storage Incubate . . . 434
Unload . . . 438
Upstack . . . 443

9. Setting parameters for liquid-handling tasks . . . 449
SubProcess (Agilent Bravo) . . . 450
Aspirate (Agilent Bravo) . . . 455
AM Aspirate (Agilent Bravo) . . . 461
AM Dispense (Agilent Bravo) . . . 471
AM Mix (Agilent Bravo) . . . 481
AM Cartridges Off (Agilent Bravo) . . . 489

AM Cartridges On (Agilent Bravo) . . . 491
AM Wash Syringes (Agilent Bravo) . . . 494
Assemble Vacuum (Agilent Bravo) . . . 499
Dilute to Final Volume (Agilent Bravo) . . . 504
Disassemble Vacuum (Agilent Bravo) . . . 512
Dispense (Agilent Bravo) . . . 514
Dispense to Waste (Agilent Bravo) . . . 521
Hit Pick Replication (Agilent Bravo) . . . 526
Initialize Axis (Agilent Bravo) . . . 558
Mix (Agilent Bravo) . . . 559
Move and Filter Plate (Agilent Bravo) . . . 568
Pin Tool (Agilent Bravo) . . . 574
Pipette Under Collar Begin and End (Agilent Bravo) . . . 584
Pump Reagent (Agilent Bravo) . . . 591
Serial Dilution (Agilent Bravo) . . . 595
Set Head Mode (Agilent Bravo) . . . 601
Set Temperature (Agilent Bravo) . . . 609
Set Temperature - Position (Agilent Bravo) . . . 611
Shake (Agilent Bravo) . . . 613
Shake Plate (Agilent Bravo) . . . 616
Tips Off (Agilent Bravo) . . . 619
Tips On (Agilent Bravo) . . . 623
Toggle Vacuum (Agilent Bravo) . . . 626
Wash Tips (Agilent Bravo) . . . 632

10. Setting parameters for scheduling tasks . . . 639
Define Plate Set . . . 640
Define Variables . . . 645
Change Instance . . . 648
Group Begin and Group End . . . 650
JavaScript . . . 653
Loop and Loop End . . . 655
Print . . . 658
Process Control . . . 666
Signal . . . 675
Spawn Process . . . 677
User Message . . . 682
Wait For . . . 686
Wait for User (Agilent Bravo) . . . 688

11. Specifying pipetting techniques . . . 691
About the Pipette Technique Editor . . . 692
Creating and editing pipetting techniques . . . 693
Managing pipetting techniques . . . 697

12. Maintenance and troubleshooting 699
Backing up and restoring VWorks files 700
Exporting and importing protocols and associated records 703
Viewing logs 714
Resolving device initialization errors 721
Resolving compilation error messages 722
Disabling and enabling tasks 723
Using breakpoints to monitor and troubleshoot tasks 724
Resolving barcode reader error messages 729
Recovering from deadlocks 730
Setting up automated error responses 735
Reporting problems 743

A. Quick reference 745
Menu commands 746
Toolbar buttons 752

B. Setting up and using the Watcher tool 755
Watcher overview 756
Creating the script that Watcher will run 757
Setting up the Watcher configuration file 761
Turning on Watcher 765

Preface

This preface contains the following topics:

- "About this guide" on page x
- "Accessing Agilent Automation Solutions user guides" on page xiii

About this guide

Who should read this guide

This user guide is for people with the following job roles:

Job role	Responsibilities
Lab manager, administrator, or technician	Someone who is responsible for: • Developing training materials and standard operating procedures for operators
Operator	Your organization may choose to create its own procedures for operators including the procedures in this guide.

What this guide covers

This guide explains how to use the VWorks Automation Control software to

- Set up devices
- Write protocols
- Run protocols
- Troubleshoot protocols
- Operate devices using VWorks software

Table Terms used in this guide

Term	Description
VWorks Automation Control	VWorks software component that you use to create the protocols that run your automation devices.
VWorks Plus	The compliance-enabled *workstation* edition of the VWorks software. This edition includes VWorks and OpenLab Control Panel, Shared Services, Secure Storage, and Content Browser installed on a single computer.
VWorks Standard	The *workstation* edition of the VWorks software that is not compliance enabled. This edition includes VWorks and OpenLab Control Panel and Shared Services installed on a single computer.
Secure Storage (OpenLab component)	The secure data storage repository for VWorks compliance-enabled editions.
Control Panel (OpenLab component)	The user interface for Shared Services.
Microsoft Control Panel	Part of the Microsoft Windows operating system.

Term	Description
Shared Services (OpenLab component)	Set of administrative services that control VWorks user access and file storage. Shared Services are accessed via the Control Panel.

Software version

This guide describes VWorks Automation Control 14.1.1 and later versions.

What is new in this edition

Updates for VWorks 14.4	See...
Replaced the term Content Management with Secure Storage.	• "Terms used in this guide" on page x • "Architecture and features overview" on page 2

Related guides

This guide should be used in conjunction with the following user documents.

For information about ...	See...
Computer requirements and installing the software	*VWorks 14.4 Installation Guide*
• Configuring the software in Control Panel, including managing licenses and user access • Backing up and restoring the software, Shared Services database, and project content	• *VWorks Plus Configuration and Administration Guide* • *VWorks Standard Configuration and Administration Guide*
Setting up a specific device and operating the device using VWorks and device diagnostics software	Relevant Agilent device user guide
• *VWorks Plus only*. Audit trails, record states, e-signatures, and tamper detection • Setting up labware definitions and liquid classes • Setting up the Experiments database and Inventory database • Migrating protocols from previous versions of VWorks software.	*VWorks Automation Control Setup Guide*

For information about …	See...
An overview of how to create and run protocols.	*VWorks Quick Guide for Protocol Authors*
Setting up a specific device and operating the device using VWorks and device diagnostics software	Relevant Agilent device user guide
General safety information and potential safety hazards that you might encounter when using Automation Solutions products	*Automation Solutions Products General Safety Guide*

Related information

For information about...	See...
Accessing related user guides	"Accessing Agilent Automation Solutions user guides" on page xiii
Reporting problems with the software	"Reporting problems" on page 743

Accessing Agilent Automation Solutions user guides

About this topic

This topic describes the different formats of user information and explains how to access it for the Agilent Automation Solutions products.

Where to find user information

The user information is available in the following locations:

- *Knowledge base*. The help system for the Automation Solutions products is available from:
  - Help menu within the VWorks software: Select **Help** > **Knowledge Base** or press F1.
  - From the Windows desktop: Select **Start ( )** > **All Apps > Agilent Technologies > VWorks Knowledge Base.**

  For guidelines on using the VWorks context-sensitive help and knowledge base features, see *Using the knowledge base*, below.
- *PDF files*. The PDF files of the user guides are installed with the VWorks software (C:\Program Files (x86)\Agilent Technologies\VWorks\UserGuides) and are available in the VWorks Knowledge Base.
- *Website*. You can search the online VWorks Knowledge Base or download the latest version of any PDF file from the Agilent website at https://automation.help.agilent.com.

Accessing safety information

Safety information for the Agilent Automation Solutions devices appears in the *Automation Solutions Products General Safety Guide* and in the corresponding device safety guide or user guide.

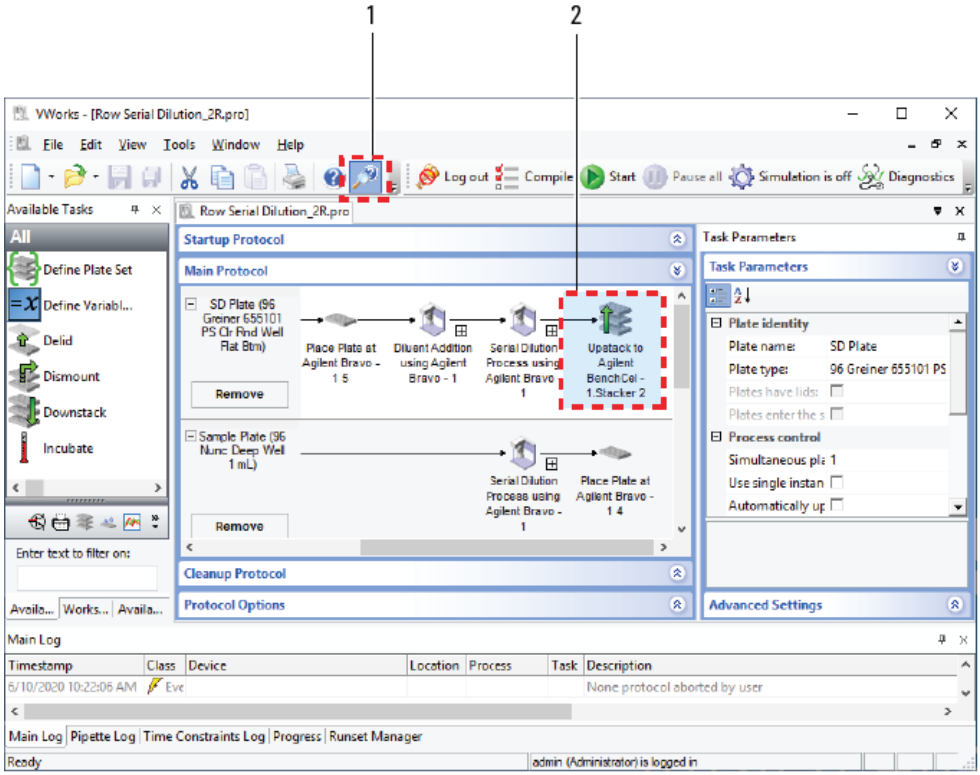
You can also search the knowledge base or the PDF files for safety information.

Using the knowledge base

Knowledge base topics are displayed using web browser software such as Microsoft Edge.

Note: If you want to use Microsoft Internet Explorer to display the topics, you might have to allow local files to run active content (scripts and ActiveX controls). For instructions, see the Microsoft user documentation.

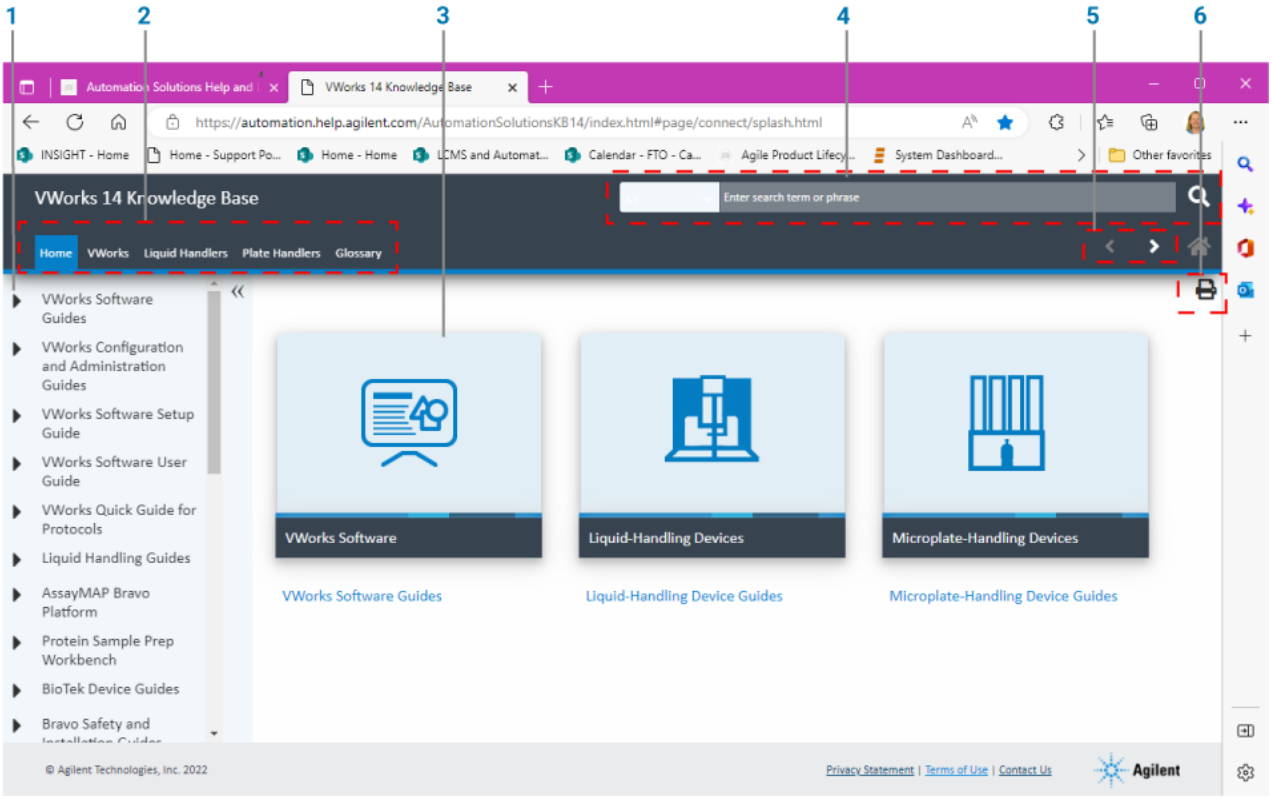
Opening the help topic for an area in the VWorks window



To access the context-sensitive help feature:

1. In the main window of the VWorks software, click the help button  . The pointer changes to  . Notice that the different icons or areas are highlighted as you move the pointer over them.
2. Click an icon or area of interest. The relevant topic or document opens.

Features in the Knowledge Base window



Item	Description
1	*Contents pane*. Lists all the books and the table of contents of the books.
2	*Tabs*. Allows you to search the Knowledge Base by product type.
3	*Topic area*. Displays content of selected tab or topic. In the above example, the content of the Home tab is displayed in the topic area.
4	*Search bar*. Enables you to search the entire knowledge base for a word or term.
5	*Navigation buttons*. Enable you to navigate through the next or previous topics listed in the Contents pane.
6	*Print button*. Enables you to print the information in the topic area.

1 Software overview

This chapter contains the following topics:

- "Architecture and features overview" on page 2
- "Supported devices" on page 9
- "Relationship of components in VWorks software" on page 5
- "Overview of VWorks user interface" on page 11

Architecture and features overview

You use the VWorks software to control your Agilent automation devices. This topic provides an overview of VWorks architecture and describes the primary features.

Architecture and primary software components

VWorks v14.1 has the following editions:

VWorks edition	Description
VWorks Plus	The compliance-enabled *workstation* edition of the VWorks software. This edition includes VWorks and OpenLab Control Panel, Shared Services, Secure Storage, and Content Browser installed on a single computer.
VWorks Standard	The *workstation* edition of the software required for running the standard VWorks software. This edition includes VWorks and OpenLab Control Panel and Shared Services installed on a single computer.

All VWorks editions consist of the following:

- Components of the OpenLab software

  Your system administrator manages VWorks user access through the OpenLab Control Panel. Shared Services stores the VWorks records. The storage type varies based on the edition.

  For details on the OpenLab for VWorks components, see the configuration and administration guide for your VWorks edition.
- VWorks software component

The following figures show the relationships between the components for each VWorks edition.

Figure VWorks Plus architecture

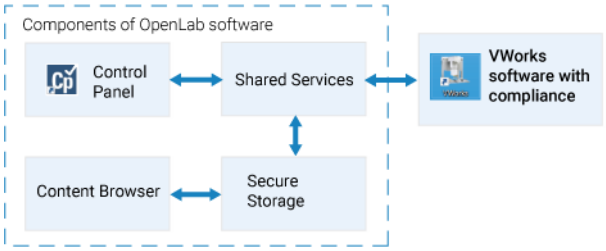
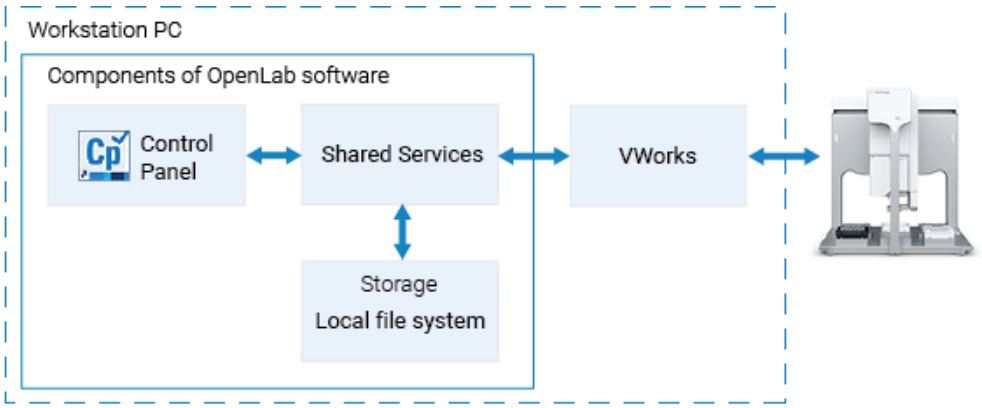


Figure VWorks Standard architecture



VWorks features

The following table lists the primary VWorks features.

Features	Description	See...
VWorks Plus only. Log audit trails, manage record states, and use e-signatures	The software logs an audit trail for each record of interest. If enabled, e-signatures are required to approve each record state transition and to close experiment IDs.	*VWorks Automation Control Setup Guide*
Define labware	You create labware definitions that describe the labware you will use during protocol runs.	*VWorks Automation Control Setup Guide*
Define liquid classes	*Bravo Platform only*. You use a liquid class to describe the liquid-handling properties for a task in a protocol.	*VWorks Automation Control Setup Guide*
Manage inventory	*Storage devices only*. For devices such as the Labware MiniHub, the Inventory Editor helps you track barcodes and labware as you move them into and out of storage or incubation.	*VWorks Automation Control Setup Guide*
Use Experiments database	You can use experiment IDs and the Experiments database to record the protocol steps performed and settings used during the course of a sample prep workflow.	*VWorks Automation Control Setup Guide*
Set up devices	You define any devices used in a protocol in a VWorks device file.	"Adding devices" on page 26
Create protocols	Protocols determine the sequence of tasks to be automated in a run. For example, you can use a protocol to apply barcode labels to 100 microplates.	"Workflow for creating a basic protocol" on page 22
Create forms	A form provides a simplified interface for running a protocol, which can be helpful for users who have limited or infrequent experience with the VWorks software.	"Creating forms for protocol operators" on page 177
Create pipetting techniques	*Bravo Platform only*. You can create pipetting techniques that describe the pipetting methods.	"Specifying pipetting techniques" on page 691

Features	Description	See...
Run, pause, monitor, and stop protocols	You can start, pause, monitor, and stop a protocol run from the controlling computer.	"Workflow for running a protocol" on page 288

Note: The VWorks software uses dynamic scheduling to run the tasks in a protocol. The scheduler determines which task to run next based on the availability of resources moment-by-moment, to reduce idle time during a protocol. This causes the software to re-determine the sequence of task execution every time a protocol is run. As a result, run times can vary for a given protocol. However, you can specify a time interval between two dependent tasks using the Time Constraints feature. For details, see "Specifying time constraints between dependent tasks" on page 71.

Related information

For information about...	See...
VWorks Plus, Control Panel, Shared Services, and Secure Storage	"Configuration and administration guides" on page 3 *VWorks Plus Configuration and Administration Guide*
VWorks Standard, Control Panel and Shared Services	"Guides for VWorks Standard edition" on page 3*VWorks Standard Configuration and Administration Guide*
VWorks file types	"Relationship of components in VWorks software" on page 5
VWorks user interface	"Overview of VWorks user interface" on page 11
Using VWorks ActiveX control	Device ActiveX control user guide

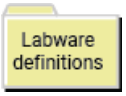
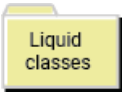
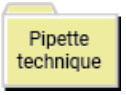
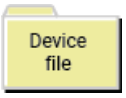
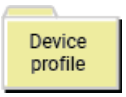
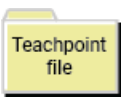
Relationship of components in VWorks software

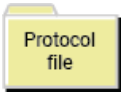
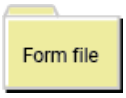
The VWorks software uses different components (files and databases) to run protocols. It is important to understand the relationships between each of the components in the software. Changing settings or options in one component will affect one or more of the other components.

Component descriptions

The following table lists and describes the primary VWorks software components.

Note: VWorks treats these as records of interest. For details on records of interest, see the *VWorks Automation Control Setup Guide*.

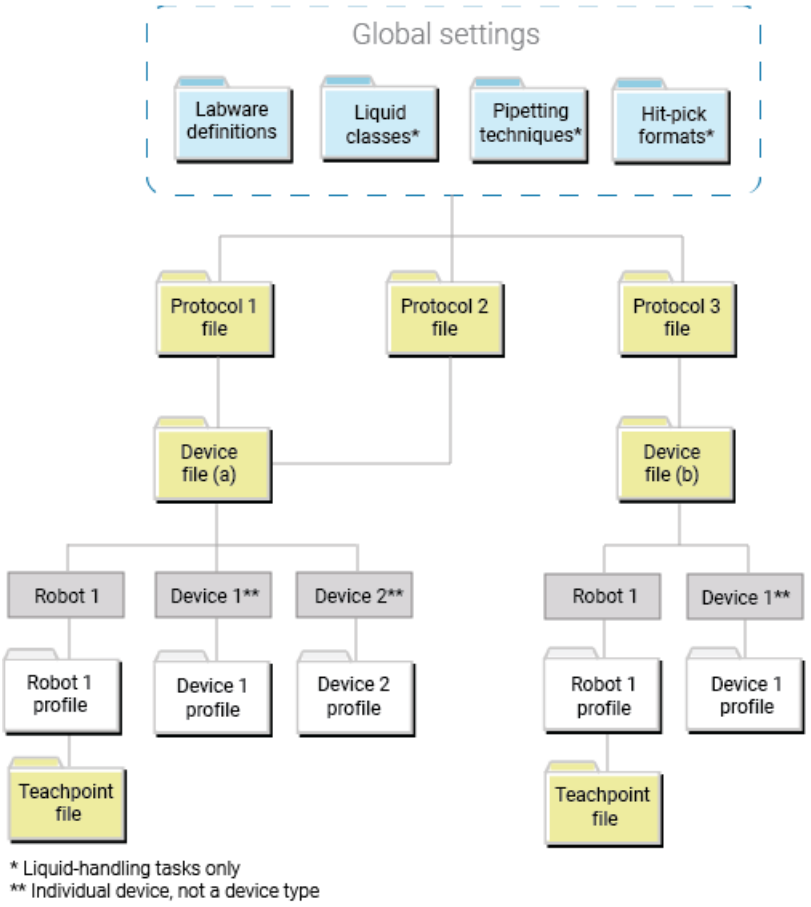
Component	Description	File extension	Opening this component loads...
Global settings			
	A file that contains a collection of labware properties that the VWorks software uses to handle a labware type in the automation system.	.xml	Labware definitions
	A file used by the Hit Pick Replication task that specifies the dispense pattern in destination microplates.	.xml	Dispense information in destination microplates
	A file that contains pipetting settings for different liquid types.	.xml	Liquid class information
	A file that specifies the *x*- and *y*-axis offset when pipetting.	.xml	Pipette *x*- and *y*-axis offset information
Device-related files			
	A file that contains: • The list of devices the software will communicate with and control • Configuration information of each device • Communication settings (profile)	.dev	• Device profile • Teachpoint file (BenchCel device)
	A file that contains a collection of settings that the VWorks software uses to control a specific device.	.xml	Teachpoints or teachpoint file (applicable to some devices such as the BenchCel device)
	A device-dependent file that contains your teachpoint settings.	.xml	Teachpoint definitions
Protocol-related files			

Component	Description	File extension	Opening this component loads...
	A file that contains instructions for performing a run.	.pro	• VWorks software (if it is not already open) • Device file • VWorks form (if configured in Protocol Options)
	A simplified user-interface that can be linked to a protocol for operators to use when running the protocol.	.VWForm	VWorks form and associated protocol

Relationships of VWorks components

The following figure summarizes the relationship of the components. Notice the following:

- Labware definitions are used by all protocols.
- Liquid classes, pipetting techniques, and hit-pick formats are used by protocols containing liquid-handling tasks. In particular, hit-pick formats are used by the Hit Pick Replication task.
- Each protocol references a single device file that contains one or more devices.
- More than one protocol can reference the same device file.
- Each device (or robot) references a single device profile.
- Some devices, such as the BenchCel device, reference a single teachpoint file



The following table describes how changes to one component impact other components.

If you...	Then...
Make a change to the teachpoint file	• All profiles that use that teachpoint file are affected • All device files that use those profiles are affected • All protocols that use those device files are affected
Create a new profile	You must specify the new profile in your device file
Want to use two different teachpoint files	You must create two: • Device files • Profiles • Teachpoint files • Protocol files

If you...	Then...
Want to copy a protocol to another system or computer	Use the **File > Export VZP** command to export all components: • Protocol file • Device file • Device profiles • Labware definitions and classes • Liquid classes • Pipette techniques • Hit-pick format or input files • Plate map database The exported .vzp file can be imported in VWorks running on another computer.
VWorks Plus only. Change the state of a record of interest that has dependencies, for example, if you change a protocol from In Development to In Validation.	The dependent records of interest must be set to an equivalent state or higher. For example, if the protocol is In Validation, the device file and other files that protocol references must also be In Validation or Released.

Miscellaneous input files

Files created in software other than the VWorks software, such as barcode input files (.bar, .csv, or .txt), hitpick input files (.csv), inventory files (.sql), JavaScript files (.js), and so forth, must be imported into Shared Services storage before the files are available for use in the VWorks software. For details, see "Importing and exporting miscellaneous files" on page 49.

Related information

For information about...	See...
VWorks software description	"Architecture and features overview" on page 2
VWorks software user interface	"Overview of VWorks user interface" on page 11
Supported devices	"Supported devices" on page 9
VWorks Plus, Control Panel, Shared Services, and Secure Storage	*VWorks Plus Configuration and Administration Guide*
VWorks Standard, Control Panel and Shared Services	*VWorks Standard Configuration and Administration Guide*
Device setup, including creating profiles and setting teachpoints	Device user guide
Labware definitions	*VWorks Automation Control Setup Guide*

For information about...	See...
Liquid classes	*VWorks Automation Control Setup Guide*
Audit trails	*VWorks Automation Control Setup Guide*
Using VWorks ActiveX control	Device ActiveX control user guide
Exporting protocols and associated components	"Exporting and importing protocols and associated records" on page 703

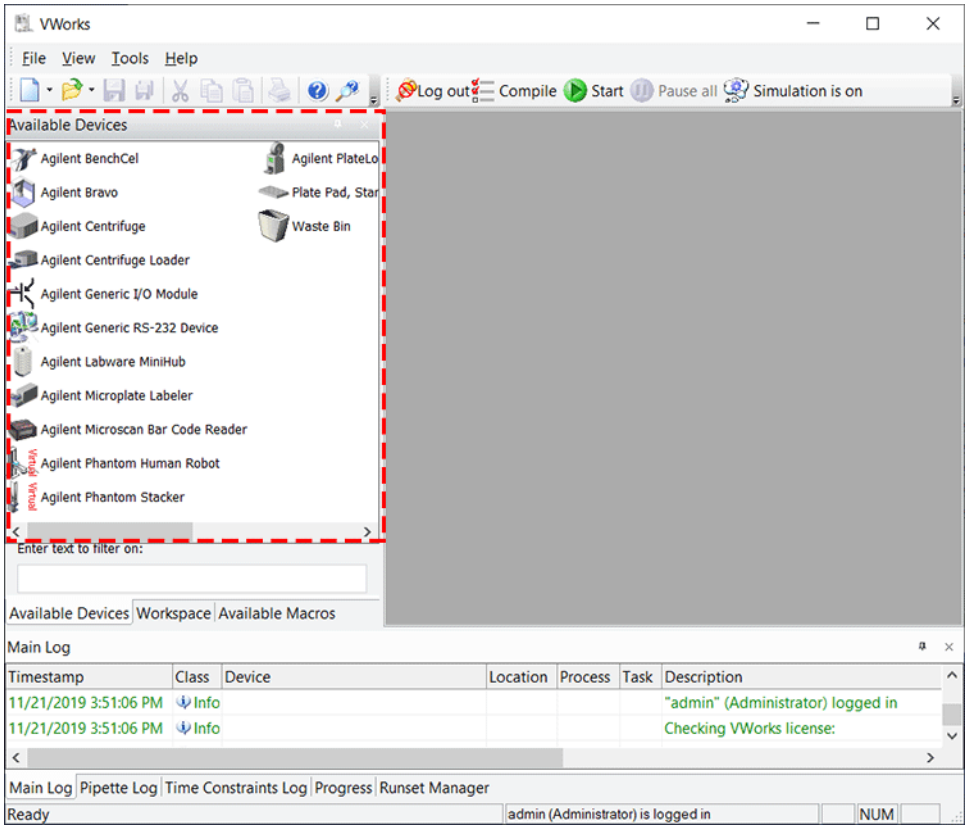
Supported devices

Devices overview

The VWorks software uses device plugins to manage and control Agilent Automation Solutions devices. A device can be a robot, an instrument, or a location in the lab automation system that can hold a piece of labware.

When you start the VWorks software, the icons of the plugins for the supported devices are listed in the Available Devices area of the window, as the following figure shows.

Figure Available Devices list in the VWorks window



Devices that VWorks supports

Device type	Description
Labware-handling devices	• Agilent Benchcel • Agilent Centrifuge • Agilent Centrifuge Loader • Agilent Labware Minihub • Agilent Microplate Labeler • Agilent Microscan Barcode Reader • Agilent PlateLoc • BioTek Gen5 Reader
Liquid-handling devices	Agilent Bravo BioTek Liquid Handler
Phantom devices	Virtual devices that permit the manual movement of labware during a protocol run: • Agilent Phantom Human Robot • Agilent Phantom Stacker
Location devices	Devices where you can place a piece of labware: • Platepad (installed outside of Bravo Platform) • Waste Bin
Devices for manufacturing use only	Agilent Generic I/O Module Agilent Generic RS-232 Device

Related information

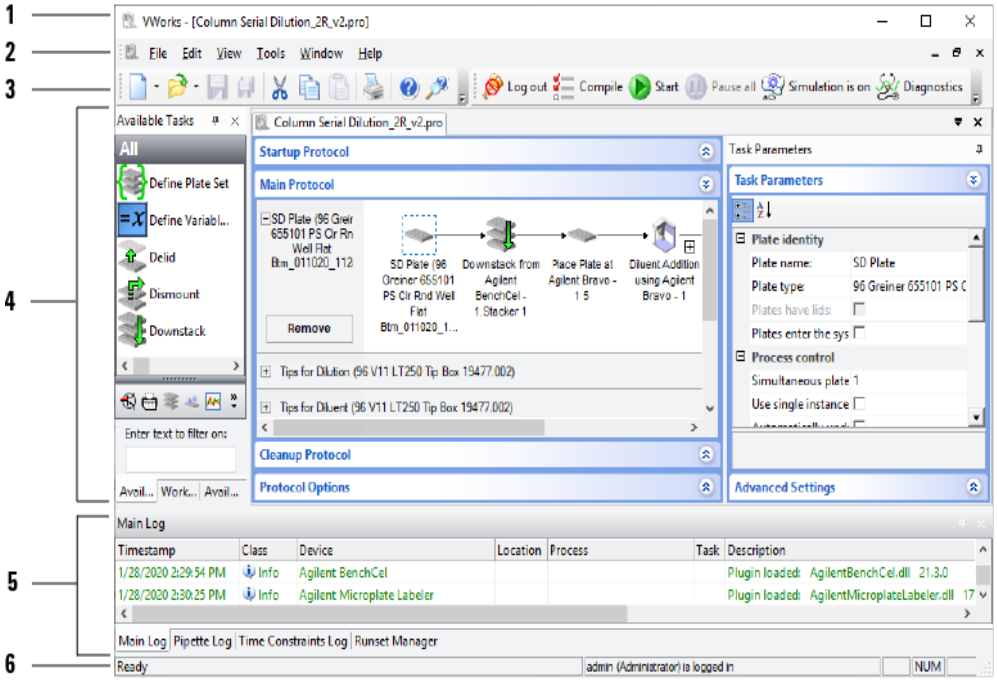
For information about...	See...
Creating a VWorks device file	"Adding devices" on page 26
How to set up specific device	Agilent device user guide
Relationship of VWorks software components	"Relationship of components in VWorks software" on page 5
Introduction to the VWorks software user interface	"Overview of VWorks user interface" on page 11
Labware definitions	*VWorks Automation Control Setup Guide*
Liquid classes	*VWorks Automation Control Setup Guide*

Overview of VWorks user interface

The VWorks user interface consists of menus and menu commands, toolbars, tabbed areas, and a status bar. The content of each of these items can change depending on whether you are viewing a device file, protocol, or form.

Basic terminology

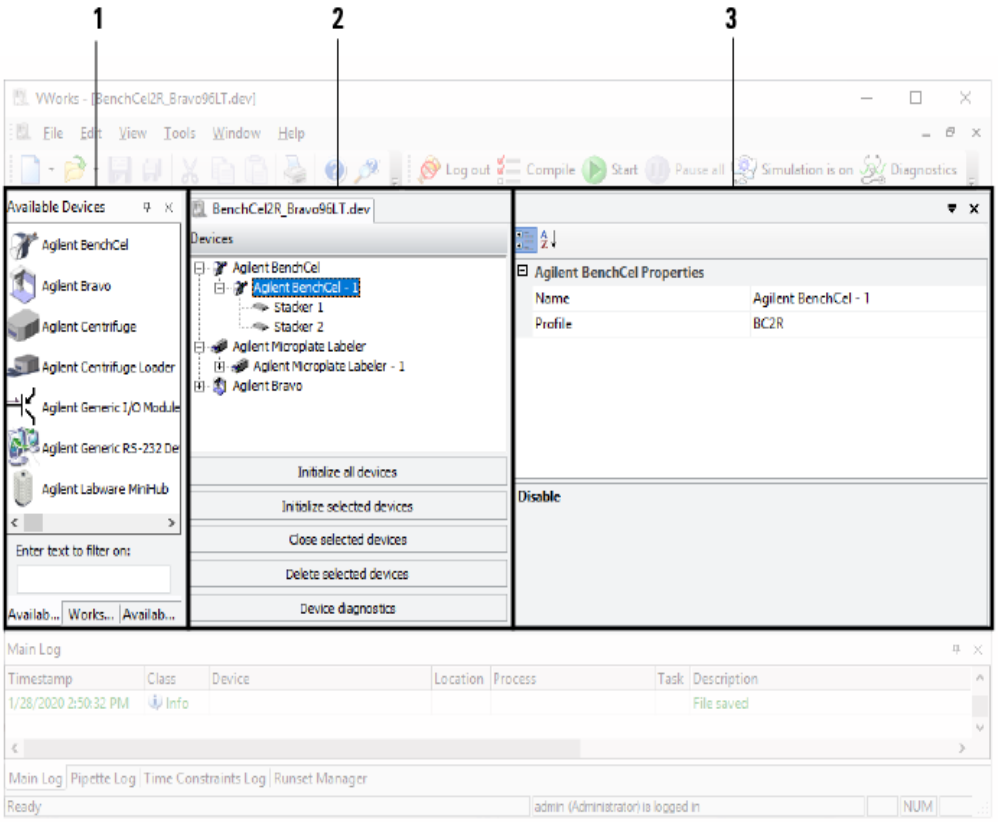
The following figure shows the basic VWorks user interface elements.



Item	Terminology	Description
1	Title bar	Displays the name of the window.
2	Menus	Lists menu commands.
3	Toolbars	Displays button commands.
4	Work area	Displays device files or protocols and forms.
5	Log and progress area	Displays the Main Log, Pipette Log, Time Constraint Log, Progress, and Runset Manager. You use these tabs primarily to monitor various aspects of the run and to set up multiple protocol runs in a sequence.
6	Status bar	Displays the state of the software.

Device file terminology

To display device file information, click the tab that displays the name of the device file. The following figure shows an example device file. By default, the software assigns a sequentially numbered device file name: Device File - 1, Device File - 2, and so forth. In the example, the device file has been renamed BenchCel2R_Bravo96LT.dev.

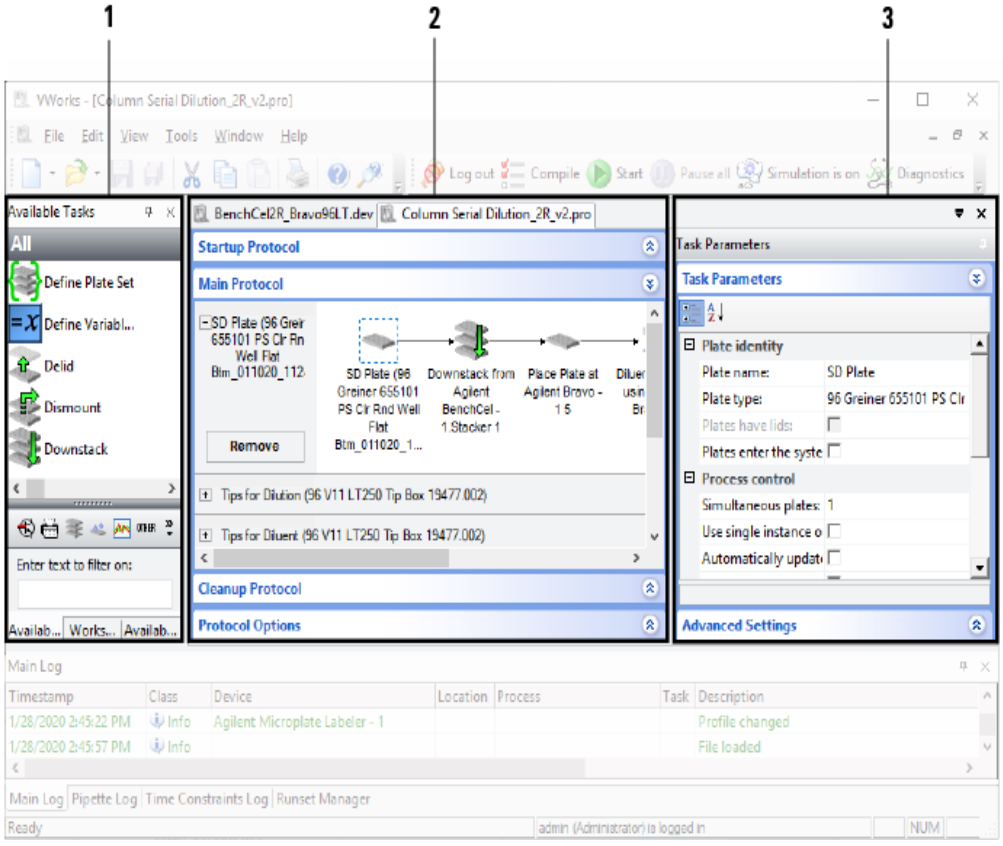


Item	Terminology	Description
1	Available Devices area	Displays the following tabs: • **Available Devices**. The list of devices you can add to the device file. • **Workspace**. The list of protocols, forms, and device files that are currently open. • **Available Macros**. The list of macros created to expedite the protocol writing process.

Item	Terminology	Description
2	Device file area	Displays the list of devices that are added to the device file. The area also allows you to: • Initialize all devices. • Initialize selected devices. • Close selected devices. • Delete selected devices. • Open the diagnostics software of the selected device.
3	Device properties area	Displays the properties you can set for the selected device in the device file.

Protocol terminology

To display the protocol information, click the tab that displays the name of the protocol. The following figure shows an example protocol. By default, the software assigns a sequentially numbered protocol name: Protocol File - 1, Protocol File - 2, and so forth. In the example, the protocol has been renamed Column Serial Dilution_2R_v2.pro.



Item	Terminology	Description
1	Available Tasks area	Displays the following tabs: • **Available Tasks**. The list of tasks you can add to a protocol. The list of tasks can vary, depending on the devices added in the device file and the process or subprocess selected. *Note:* You can use the filter buttons beneath the tasks to display only the tasks in a selected category. • **Workspace**. The list of protocols, forms, and device files that are currently open.
2	Protocol area	Displays the following: • **Startup Protocol**. Contains tasks that must be run before the main protocol starts. • **Main Protocol**. Contains tasks of the protocol. • **Cleanup Protocol**. Contains tasks that must be run after the main protocol finishes. • **Protocol Options**. References the device file and contains additional information associated with the protocol. An open VWorks form will display in the Protocol and Task Parameters areas. If the form displays in full screen mode, the entire form fills the VWorks window. **IMPORTANT** The computer display must be set to a scale of 100% in order for the VWorks form to display properly.
3	Task Parameters area	Displays one or more of the following, depending on the task selected: • **Task Parameters**. Contains parameters associated with the selected process plate or task. • **Custom Parameters**. Allows you to create variables to be used by the selected task. Might also contain parameters or options associated with the task. • **Device Selection**. Contains the list of devices that will perform the selected task. Also allows you to set up a pool of devices for the same task to increase throughput and backup devices in case the primary devices are in an error state. • **Advanced Settings**. Allows you to add JavaScript to change the task parameters or pass information to and from an external database during a protocol run.

Related information

For information about...	See...
Menu and toolbar commands	"Quick reference" on page 745
Supported devices	"Supported devices" on page 9
Creating a protocol	"Workflow for creating a basic protocol" on page 22
Creating a form	"Creating forms for protocol operators" on page 177

2 Creating a protocol: basic procedure

Read this chapter if you are an administrator or technician who writes protocols.

This chapter contains the following topics:

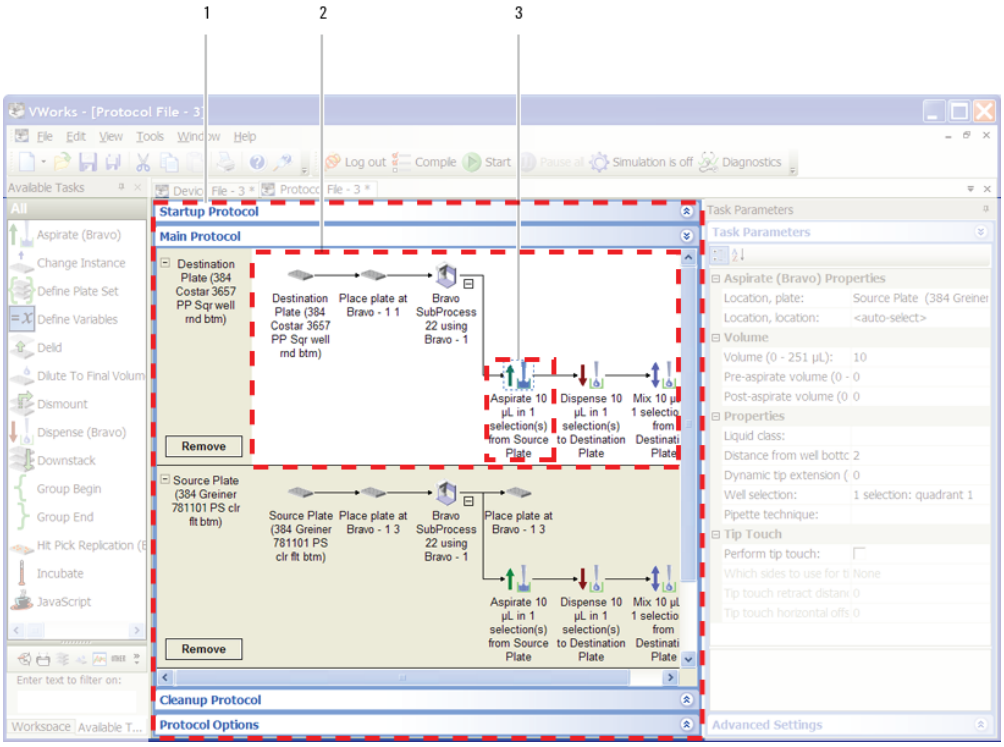
- "About protocols, processes, and tasks" on page 18
- "Workflow for creating a basic protocol" on page 22
- "Preparing for protocol writing" on page 24
- "Adding devices" on page 26
- "Setting up location devices" on page 32
- "Setting up phantom devices" on page 36
- "Creating a new protocol" on page 41
- "Setting protocol options" on page 42
- "Importing and exporting miscellaneous files" on page 49
- "Configuring labware" on page 53
- "Adding processes" on page 58
- "Setting plate parameters" on page 59
- "Adding and deleting tasks" on page 67
- "Specifying time constraints between dependent tasks" on page 71
- "Setting up Startup and Cleanup Protocol processes" on page 75
- "Saving the protocol" on page 76
- "Compiling the protocol" on page 77
- "Simulating the protocol run" on page 80
- "Printing protocols" on page 86

For details on using macros to help create your protocols, see "Using macros to create protocols" on page 265.

About protocols, processes, and tasks

Concept overview

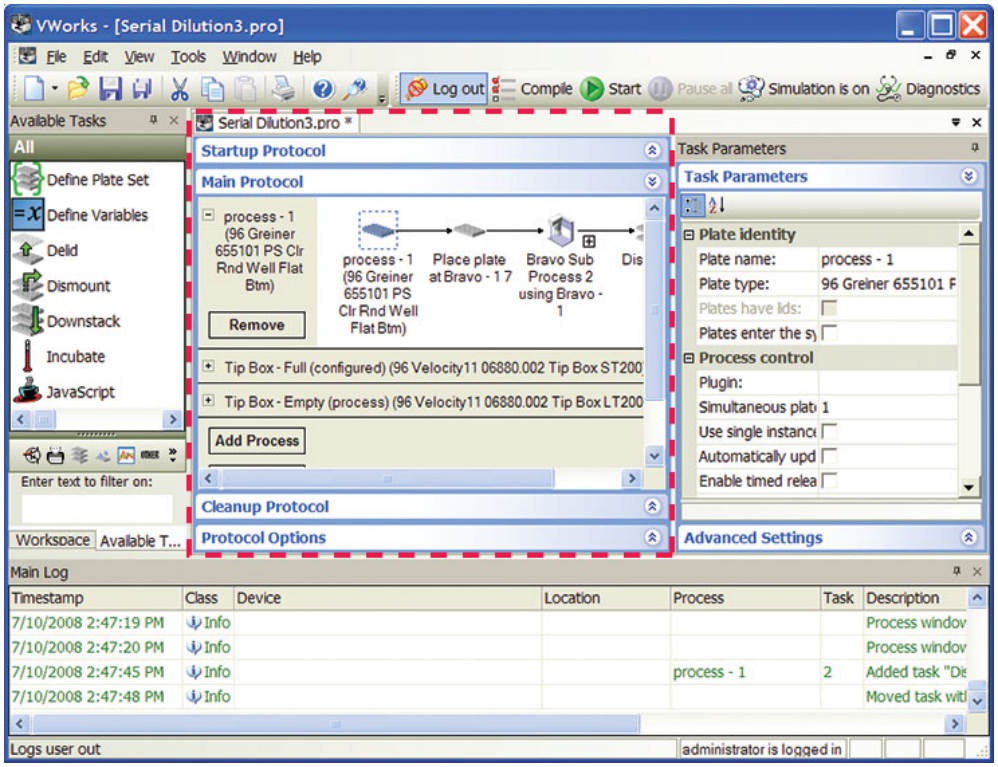
The VWorks software enables you to create protocols that automate laboratory tasks. A protocol (1) consists of one or more processes (2). Each process consists of one or more tasks (3).



Protocols

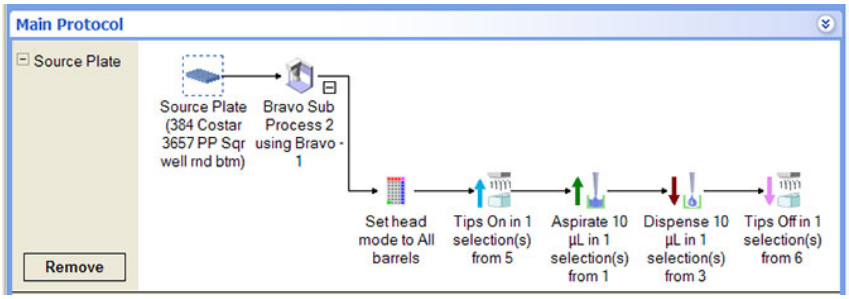
A protocol is a schedule of tasks to be performed by a standalone device or devices integrated in the lab automation system. The purpose of a protocol is to process or perform tasks on labware.

Protocols appear in the Protocol area of the VWorks window. The following example shows a protocol called Serial Dilution3 displayed in the VWorks window.



Processes

A process is a sequence of tasks that are performed on a particular labware or a group of labware. A process is represented by a lane with white background in the Protocol area.



Each process lane starts with a microplate icon. The icon typically represents the labware or group of labware that you are processing. The labware is called the process plate.

In the previous example, the process plate is called Source Plate and is associated with a specific labware type (the type is defined in the Labware Editor). The process contains a sequence of tasks for processing the Source Plate: Set head mode, Tips On, Aspirate, Dispense, and Tips Off. (For a description of the tasks, see "Setting parameters for liquid-handling tasks" on page 449.)

A process lane can also be used as a control to initiate other processes in the protocol. In this case, the microplate icon at the beginning of the process lane is not associated with any labware.

In the following example, the process lane named Control is used to initiate other processes in the protocol. Notice that the protocol can contain more than one process.

The example also shows a lane with gray background. The gray lane displays activities of configured labware. For more information about configured labware, see "Configuring labware" on page 53.

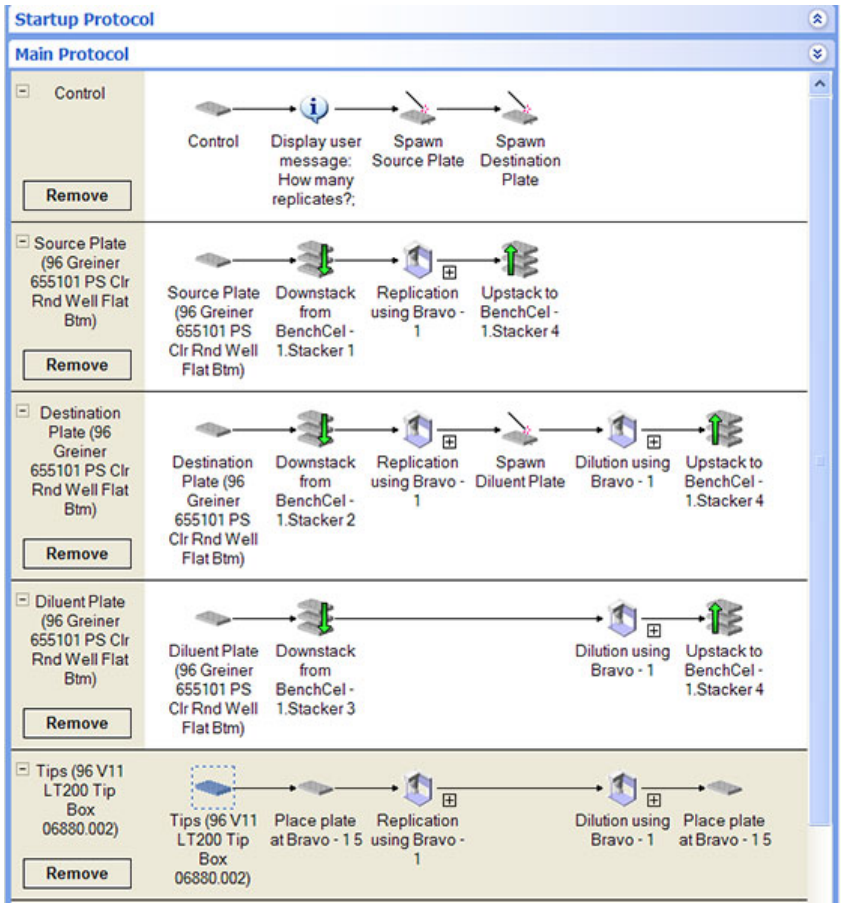


Plate instances

When a process plate icon represents a group of labware, each labware in the group is called a plate instance. Using the previous example, if you have 10 microplates that need to be processed as the Source Plate, then microplate 1 is Source Plate instance 1, microplate 2 is Source Plate instance 2, microplate 3 is Source Plate instance 3, and so on.

IMPORTANT

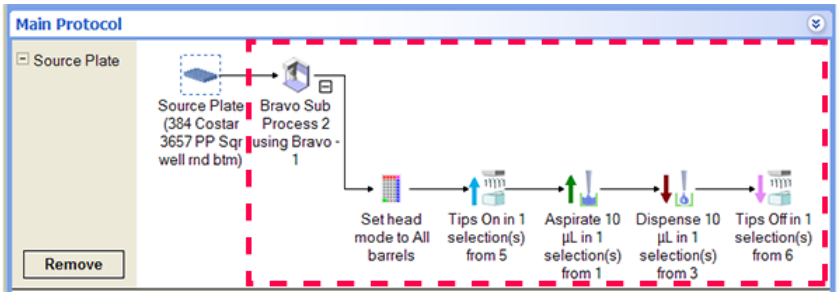
All labware represented by a process plate must be the same labware type.

Subprocesses

A subprocess is a sequence of tasks performed as a subroutine within a protocol. A subprocess is performed by a single device type, such as the Bravo device.

IMPORTANT Within a given protocol, ensure that any main process and subprocess do not share the same name.

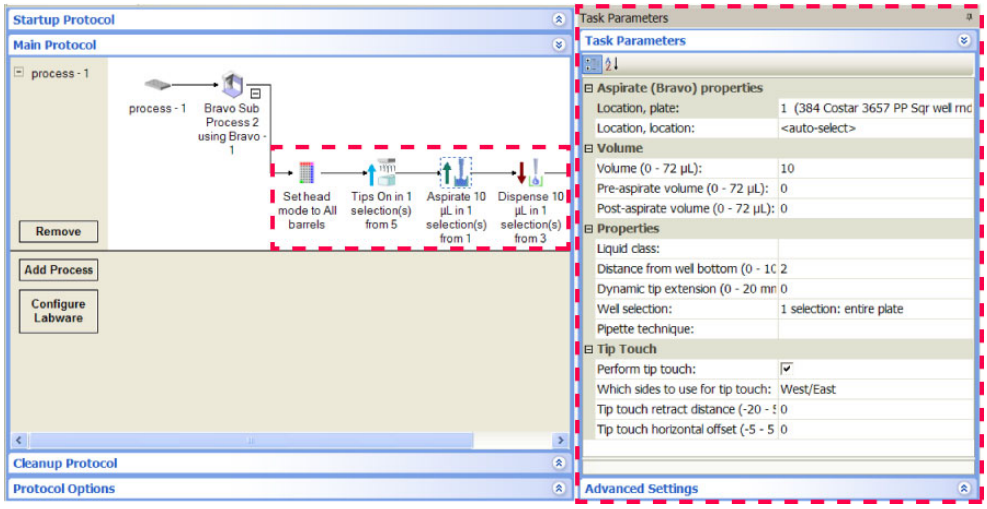
Subprocesses are represented by a subprocess icon in the protocol. You can expand or collapse the subprocess to show or hide the subprocess tasks. In the following example, the Bravo Subprocess is expanded to show the following tasks: Set Head Mode, Tips On, Aspirate, Dispense, and Tips Off. (For a description of the tasks, see "Setting parameters for liquid-handling tasks" on page 449.)



Tasks

A task is an operation performed on one or more labware, and is represented by an icon in the protocol. It has associated parameters that are set in the Task Parameters area.

In the following example, the tasks in the Bravo subprocess are highlighted. Notice that the Aspirate task is selected, and the parameters for the Aspirate task are displayed in the Task Parameters area on the right.



Related information

For information about...	See...
Defining labware	*VWorks Automation Control Setup Guide*
Preparing for protocol writing	"Preparing for protocol writing" on page 24

For information about...	See...
Workflow for creating basic protocol	"Workflow for creating a basic protocol" on page 22
Configuring labware	"Configuring labware" on page 53
Using advanced features	"Creating a protocol: advanced topics" on page 89

Workflow for creating a basic protocol

The following table presents the workflow for creating a protocol.

Step	For this task...	See...
1	Prepare for protocol writing.	"Preparing for protocol writing" on page 24
2	Log in to the VWorks software.	*VWorks Automation Control Setup Guide*
3	Add devices and create a profile for each device.	"Adding devices" on page 26
4	Create a protocol.	"Creating a new protocol" on page 41
5	Set protocol options.	"Setting protocol options" on page 42
6	If the protocol uses files created in other software, such as barcode input files, determine if you need to import the files into Shared Services storage.	"Importing and exporting miscellaneous files" on page 49
7	Configure labware.	"Configuring labware" on page 53
8	Add processes.	"Adding processes" on page 58
9	Set plate properties.	"Setting plate parameters" on page 59
10	Add tasks.	"Adding and deleting tasks" on page 67 To use macros for commonly used task sequences, see "About protocol macros and the macro library" on page 266.
11	*Optional*. Set up startup and cleanup protocols.	"Setting up Startup and Cleanup Protocol processes" on page 75
12	Save the protocol.	"Saving the protocol" on page 76
13	Compile the protocol.	"Compiling the protocol" on page 77

Step	For this task...	See...
14	Simulate the protocol run.	"Simulating the protocol run" on page 80
15	*Optional*. Print the protocol.	"Printing protocols" on page 86

Related information

For information about...	See...
Creating a simplified interface for operators to use	"Creating forms for protocol operators" on page 177
Protocols, processes, and tasks	"About protocols, processes, and tasks" on page 18
Using advanced features	"Creating a protocol: advanced topics" on page 89

Preparing for protocol writing

Before you create a protocol, determine the following:

- The devices and accessories you need for the protocol.

  A device is a robot, instrument, or location in the lab automation system that can hold a piece of labware. For more information, see "Adding devices" on page 26.

  An accessory is an option that can be added to a robot, instrument, or location to enhance existing functions and facilitate operation (for example, an Auto Filling Reservoir).
- The labware that will be used or processed during the protocol run and their starting and ending locations.
- Whether the macro library contains any task sequences that you can use. For details on using macros, see "Using macros to create protocols" on page 265.
- Whether you will use a VWorks form to run the protocol. For details, see "Creating forms for protocol operators" on page 177.

This topic explains how you plan for and specify the different devices, accessories, and labware in a protocol and the terminology that is used.

Planning device and accessory use

When planning for devices and accessories:

- Determine the devices and accessories you will need in the protocol.
- Determine the locations of the accessories on devices such as the Bravo Platform. You do this when you configure the accessory in the diagnostics software. For instructions, see the device user guide.

IMPORTANT

Accessory locations are displayed in the diagnostics Configuration tab only. You need to remember their locations when configuring labware in the protocol.

Note: Devices and accessories stay at the same location throughout a protocol run.

Planning labware use

In a lab automation system, labware can either:

- Transfer into and out of the system for processing.
- Start and stay in the system during the entire protocol run.

When you plan a protocol, you should determine how the labware will be used or processed and how they will move in the system. For example, if you are writing a microplate replication protocol, you need to decide whether the source microplates or destination microplates will be moved into the system while the other will remain stationary in the system. The decision can depend on many factors, including your preferences.

In the VWorks software, labware can be categorized as follows:

- Process plates
- Configured labware
- Static labware

Process plates

A process plate is a labware that:

- Is transferred into the system automatically or manually during the protocol run.
- Is the object of one or more tasks in the protocol run.
- Might move to different locations during the run.
- Is transferred out of the system automatically or manually during the protocol run.

For more information about process plates, see "About protocols, processes, and tasks" on page 18. To create a protocol process, see "Workflow for creating a basic protocol" on page 22.

Configured labware

A configured labware is a labware that:

- Starts at a location on a device.
- Is used by one or more tasks in the protocol process.
- Might move to different locations during the run.
- Returns to the original location after the protocol run is finished.

For the Bravo Platform, configured labware represents the single physical labware on the deck, such as a tip box. For example, if you are using two different tip boxes in a protocol, you would configure two labware in the software, one for each tip box on the deck.

Similar to accessories, you must let the software know the labware starting location. To do this, see "Configuring labware" on page 53.

Configured labware is displayed in a protocol with a gray background. If it is used by a task in a Bravo sub-process, a copy of the sub-process tasks are shown next to the configured labware. You cannot add or remove tasks in the duplicated process. However, whenever the sub-process is updated, the duplicate copy is also updated automatically.

You have the option of converting a configured labware into a process plate. For instructions, see "Configuring labware" on page 53.

Static labware

A static labware is a labware that will start on the Bravo deck and will remain at the same location during the protocol run. For example, a tip box can be a static labware.

IMPORTANT

Although the VWorks software supports the static labware configuration in protocols for backward compatibility, Agilent Technologies recommends that you use the concept and procedure in "Configuring labware" on page 53 when writing new protocols.

To specify its starting location, you must configure the static labware using one of the following methods:

- The Bravo Sub-Process task in a Main Protocol
- The Configure Static Labware task in the Startup Protocol
- In general, you configure static labware before the first task in a protocol. If you have multiple processes in the protocol, configure the labware once before the first task of the first process.

Configure static labware in a Startup Protocol if it will be used in all the Main Protocol subprocesses. Configure static labware in the Main Protocol subprocess if you want to override the labware configuration in the Startup Protocol.

Related information

For information about...	See...
Protocols, processes, and tasks	"About protocols, processes, and tasks" on page 18
Workflow for creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using advanced features	"Creating a protocol: advanced topics" on page 89
Using macros to expedite protocol writing	"Using macros to create protocols" on page 265
Creating a custom interface for the operator who runs the protocol	"Creating forms for protocol operators" on page 177

Adding devices

About devices

In the VWorks software, a device is an item in your lab automation system that can be added to the VWorks device file. A device can be a robot, an instrument, or a location in the lab automation system that can hold a piece of labware. The following are examples of devices:

- BenchCel Microplate Handler
- Bravo Automated Liquid Handling Platform
- Centrifuge with Loader
- Labware MiniHub
- Microplate Labeler
- Phantom Robot
- PlateLoc Thermal Microplate Sealer
- Platepad
- Waste Bin

About device files

To communicate with and control the robot and integrated devices, the VWorks software uses a device file that contains the following information:

- List of devices the software will communicate with and control

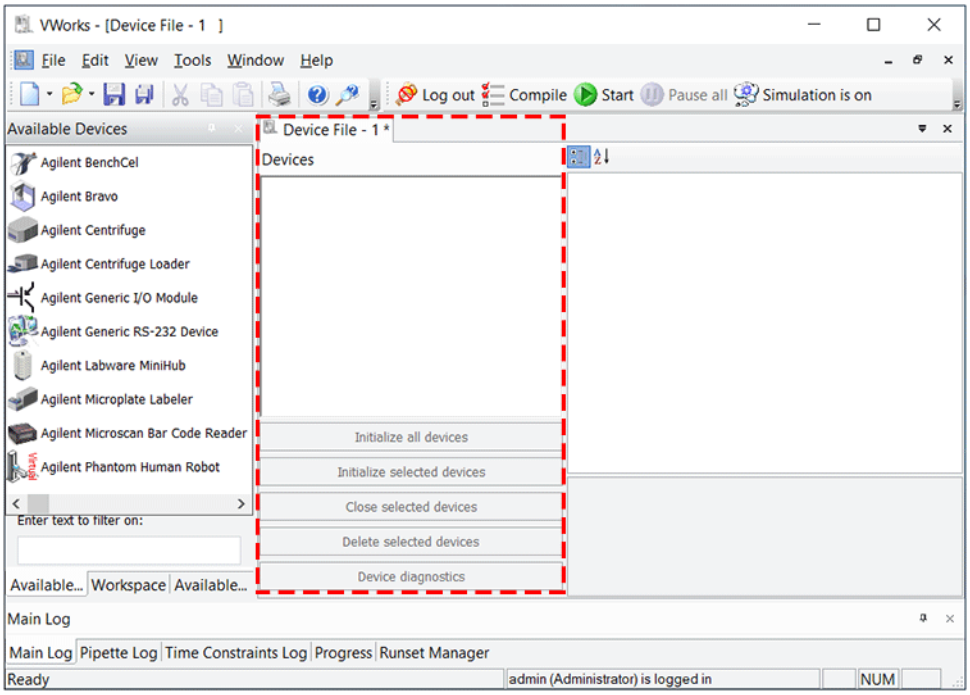
- Device type of each device (for example, the robot in the lab automation system and any integrated device)
- Configuration information of each device (for example, approach height, allowed or prohibited labware, barcode reader access, and so on)
- The communication settings (profile) required for communication between the devices and the VWorks software

You provide the device information in the VWorks window. The device information is stored in a device (.dev) file.

Creating a device file

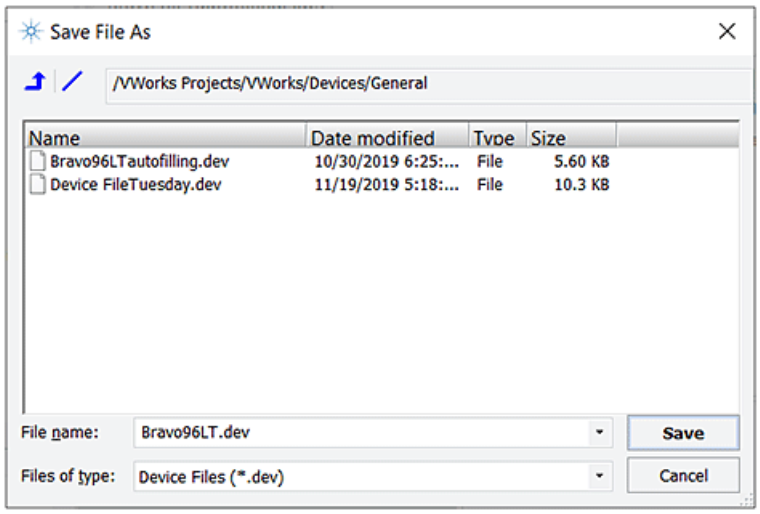
To create a device file:

1 In the VWorks window, select **File > New > Device.** A Device File tab appears.



2 Select **File > Save** to save the device file.

3 In the **Save File As** dialog box, type the file name, and then click **Save**.



4 *VWorks Plus only*. If an audit trail is being logged for records in the development state, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

Adding devices to the device file

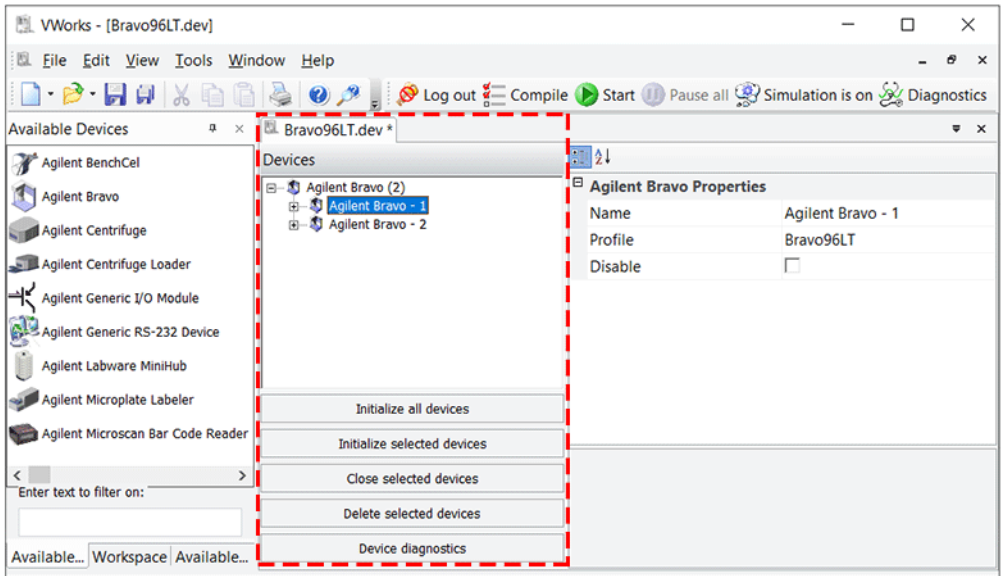
To add a device to the device file:

1 In the **Available Devices** area, double-click the device that you want to add. Alternatively, you can drag a device from the **Available Devices** area into the **Device File** area.

If you do not see the device in the **Available Devices** list, check that the device plugin file is stored in the …\Agilent Technologies\VWorks\Plugins folder.

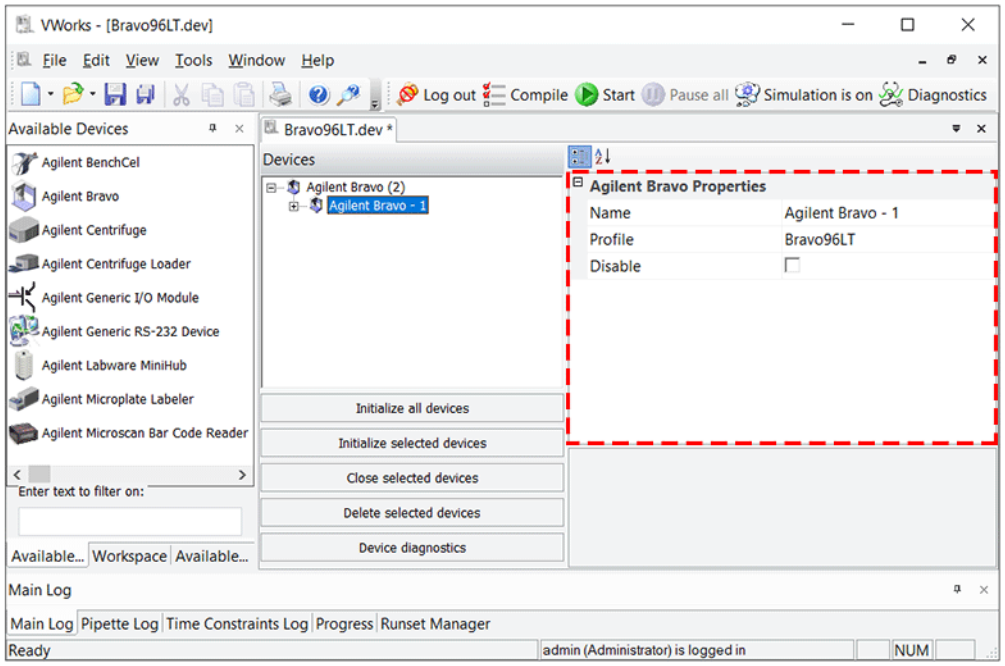
If you added a device plugin file in the Plugins folder and you have already started the VWorks software, be sure to reload the plugin. To do this, close any open device files and protocol files, and then select **Tools > Reload Plugins.**

In the following example, the Agilent Bravo device type is added. Notice that the first Bravo device is labeled Agilent Bravo-1. If you add another Bravo device to this device file, it will appear as Agilent Bravo-2.



2 Type a name for the device and set the device properties. For detailed description of the properties, see the device user guide.

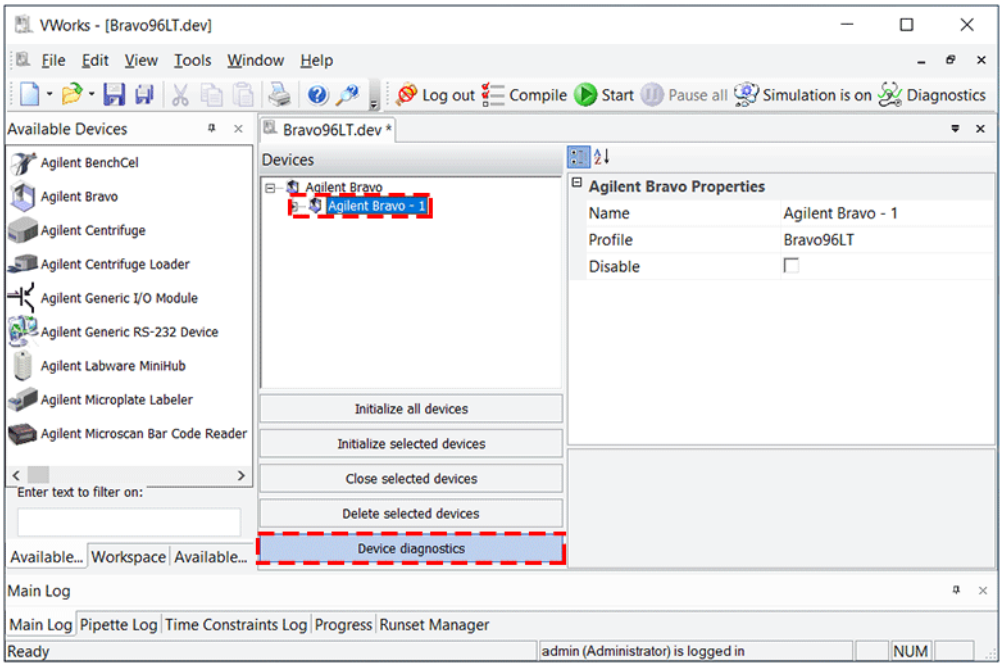
In the following example, Agilent Bravo-1 is the default name of the Bravo device. The only property shown is the profile selection.



3 Create a profile for the device:

a Select the device in the **Devices** list.

b Click **Device diagnostics.**



c In the device diagnostics dialog box, name the profile, select the connection type (Ethernet or serial), and locate and connect to the device in the Discovered Bionet Devices dialog box (Ethernet connections only).

d For devices such as the Bravo Platform and the BenchCel Microplate Handler, set the teachpoints.

For detailed instructions on creating device profiles and setting teachpoints for a specific device, see the applicable Agilent device user guide.

4 Select the profile in the device properties area.

5 Select **File > Save** to save the device file.

6 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

7 Repeat steps 1 through 5 to add other devices.

8 In the **Device File** area, click **Initialize all devices** to establish communication with the devices.

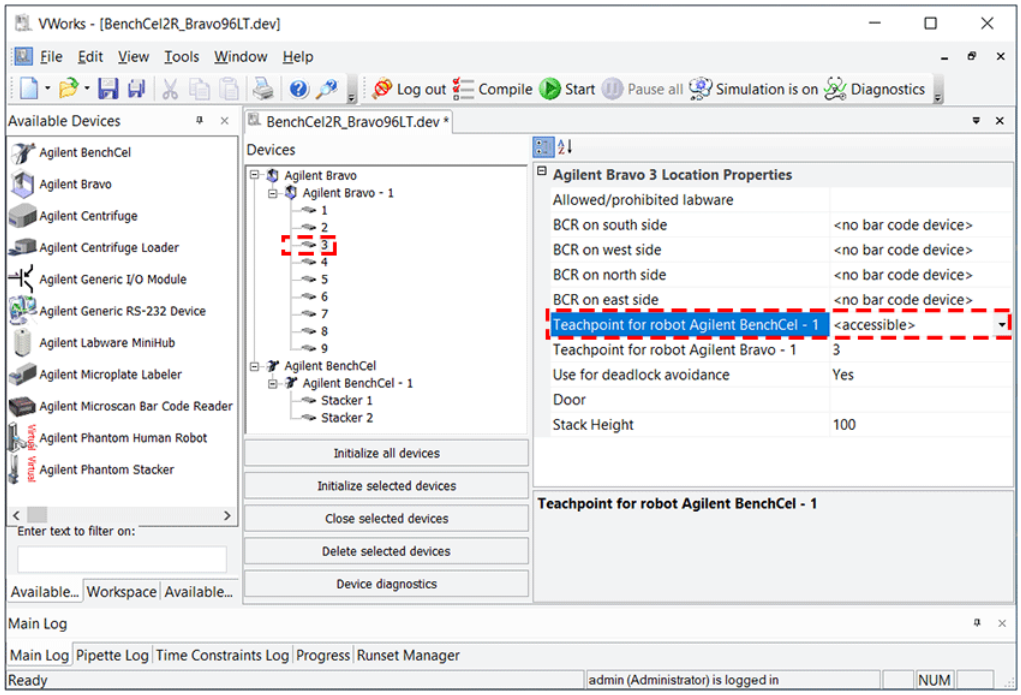
9 If you are adding devices to the BenchCel Microplate Handler, enable the robot to move to the correct locations during a protocol run as follows:

a In device diagnostics, for example, BenchCel Diagnostics, set teachpoints at each device or location. Save and reference the teachpoint file in the diagnostics dialog box.

For detailed instructions, see the user guide for the device, for example, the *BenchCel Microplate Handler User Guide*.

b In the VWorks window, select each of the devices in the device file, and select the correct robot teachpoint in the device properties area.

The following example shows a device file for a BenchCel workstation. To ensure that the BenchCel robot will move correctly to and from Bravo deck location 3, **Bravo - location 3** is selected in the **Device File** area, and the **Teachpoint for robot Agilent BenchCel-1** property specifies **<accessible>**.



10 Select **File > Save** to save the changes.

11 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

12 In the **Device File** area, click **Initialize all devices.**

Disabling devices in a device file

You can temporarily disable a device in a device file to run protocols using only a subset of devices or during troubleshooting.

For more information see, "Disabling and enabling a device in the device file" on page 293.

Related information

For information about...	See...
Device-specific properties	Applicable Agilent device user guide
Platepad and Waste Bin devices	"Setting up location devices" on page 32
Phantom devices (human robot)	"Setting up phantom devices" on page 36
Creating a new protocol	"Workflow for creating a basic protocol" on page 22

Setting up location devices

About this topic

Because a location that can be accessed by the robot requires a teachpoint, the location is treated as a device in the software.

This topic describes how to set up the following location devices:

- "Plate Pad, Standard" on page 32
- "Waste Bin" on page 34

Plate Pad, Standard

Description

The standard platepad is a metal block on which a single microplate sits. Raised tabs on each side keeps the microplate in place.

Setup workflow

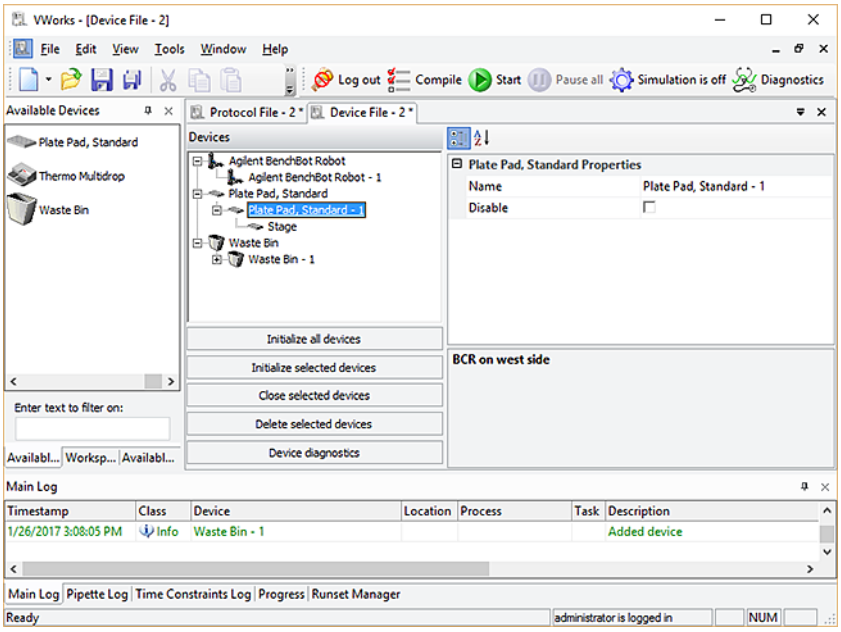
Step	For this task...	See...
1	Install the platepad.	Agilent Technical Support
2	Add the device to the VWorks device file.	"Adding the platepad in the device file" on page 32
3	Set and verify the teachpoint for the platepad.	Applicable device user guide, such as the *BenchCel Microplate Handler User Guide*

Adding the platepad in the device file

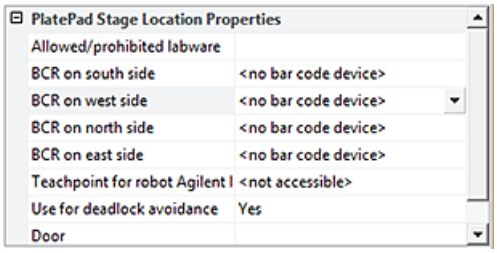
To place a microplate on or remove a microplate from a standalone platepad, you must add the Plate Pad, Standard device in the device file.

To add a platepad to the device file:

1. In the device file, add the Plate Pad, Standard device. See "Adding devices" on page 26 for instructions.
2. In the **PlatePad Standard Properties** area, type the name of the platepad.
3. In the **Devices** area, expand the **Plate Pad**, **Standard** icon and then select the **Stage** location.



4 In the **PlatePad Stage Location Properties** area, type or set the following:



Property	Description
Allowed/prohibited labware	Permitted labware class for the selected location. Use this field only if you want to limit the labware types. For details on labware classes, see the *VWorks Automation Control Setup Guide*.
BCR on south/west/ north/east side	The location of the barcode reader and the desired barcode reader device. Use this field only if a barcode reader is installed on the device.
Teachpoint for robot	The name of the teachpoint to use for the stage location. This selection enables the robot to move correctly to and from the platepad or device during a protocol run. For example, if the robot teachpoint file contains a teachpoint for this platepad, you must select that teachpoint.

Property	Description
Use for deadlock avoidance	Option to permit the location to be used for deadlock avoidance. Select Yes to permit labware to be moved to this location to avoid a deadlock in the system. Select No if you do not want to move random labware to this location to avoid deadlock.
Door	Not applicable

5 Select **File > Save**.

6 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

Waste Bin

Description

The waste bin is a container that receives waste in the automation workstation. The software views the waste bin as a location (teachpoint) where the robot releases the labware that it has in its gripper. A receptacle for trash is typically positioned under this teachpoint.

If you are configuring a waste location on the Bravo deck, see the *Bravo Platform User Guide* for instructions.

Setup workflow

Step	For this task...	See...
1	Install the waste bin.	Agilent Technical Support
2	Add the Waste Bin device to the device file.	"Adding the waste bin in the device file" on page 34
3	Set and verify the teachpoint for the location.	Applicable device user guide, such as the *Bravo Platform User Guide*

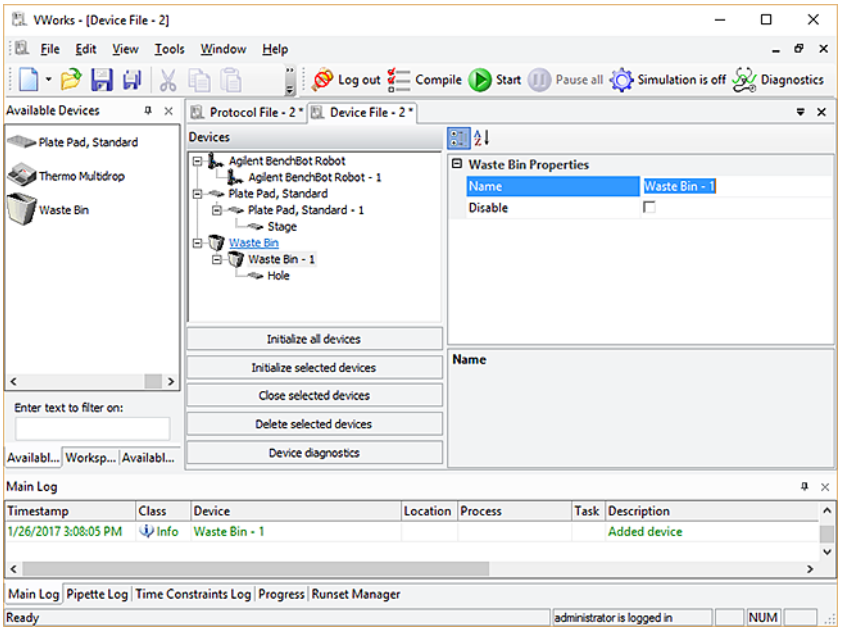
Adding the waste bin in the device file

To add a waste bin to the device file:

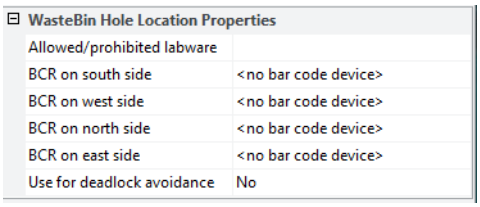
1 In the device file, add the Waste Bin device. See "Adding devices" on page 26 for instructions.

2 In the **Waste Bin Properties** area, type the name of the waste bin.

3 In the **Devices** area, expand **Waste Bin - 1**, and then select **Hole**, the location where the robot will drop waste.



4 In the **Waste Bin Hole Location Properties** area, type or set the following:



Property	Description
Allowed/prohibited labware	Permitted labware class for the selected location. Use this field only if you want to limit the labware types. For details on labware classes, see the *VWorks Automation Control Setup Guide*.
BCR on south/west/north/east side	The location of the barcode reader and the desired barcode reader device. Use this field only if a barcode reader is installed on the device.
Teachpoint for robot	The name of the teachpoint to use for the Hole location. This selection enables the robot to move correctly to and from the Waste Bin during a protocol run. For example, if the robot teachpoint file contains a teachpoint for this Waste Bin, you must select that teachpoint.
Use for deadlock avoidance	Not applicable.
Door	Not applicable.

5 Select **File > Save**.

6 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

Related information

For information about...	See...
Device files	"Adding devices" on page 26
Phantom devices	"Setting up phantom devices" on page 36
Other devices	Device user guide
Creating a new protocol	"Workflow for creating a basic protocol" on page 22

Setting up phantom devices

About this topic

In the VWorks software, every task is associated with a device. A phantom device permits the manual movement of labware during a protocol run.

This topic describes how to set up the following phantom devices:

- Agilent Phantom Human Robot
- Agilent Phantom Stacker

Agilent Phantom Human Robot

Description

The Phantom Robot is the virtual device that represents the operator who will move labware manually during a programmed pause in a protocol run. The software displays a message during the pause. You can indicate the completion of the task in the message dialog box so the protocol can continue. For a use case of the Phantom Human Robot, see the example for "Print" on page 658.

Setup workflow

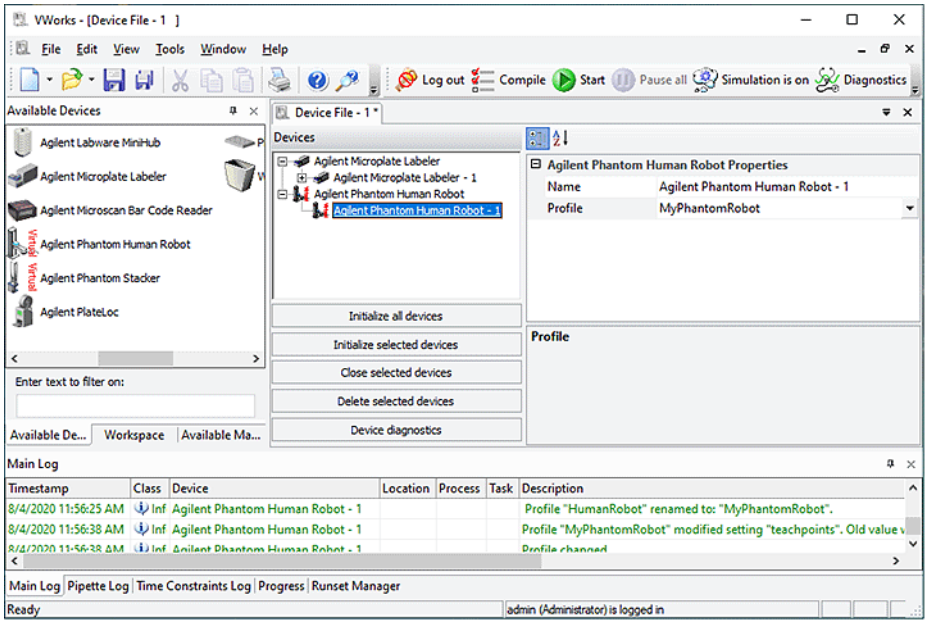
Step	For this task...	See...
1	Add the Agilent Phantom Human Robot in the device file.	"Adding the Agilent Phantom Human Robot in the device file" on page 36
2	Create a profile for the Agilent Phantom Human Robot.	"Creating a profile for the Agilent Phantom Human Robot" on page 37

Adding the Agilent Phantom Human Robot in the device file

To add the Agilent Phantom Human Robot to the device file:

1 In the device file, add the Agilent Phantom Human Robot. See "Adding devices" on page 26 for instructions.

2 In the **Devices** area, select the device. In the **Agilent Phantom Human Robot Properties** area, type or select the following:



Property	Description
Name	The name of the Phantom Human Robot.
Profile	The profile associated with the device. Select the desired profile from the list. If you have not created a profile, see "Creating a profile for the Agilent Phantom Human Robot" on page 37, and then return to this step to select the profile.

3 Select **File > Save**.

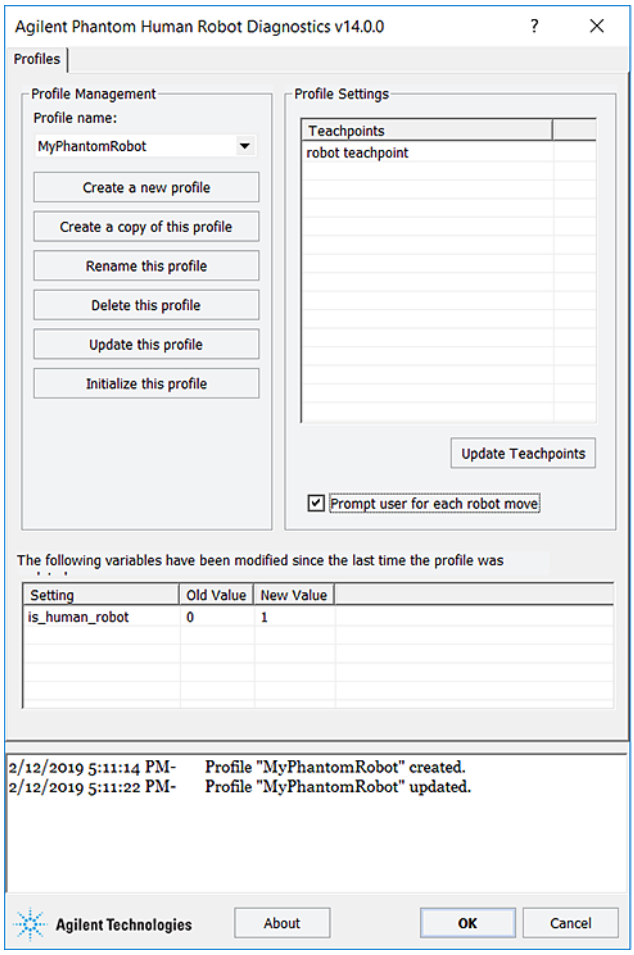
4 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

Creating a profile for the Agilent Phantom Human Robot

To create a profile for the Agilent Phantom Human Robot:

1 In the VWorks window, select the **Agilent Phantom Human Robot** device in the **Devices** area, and then click **Device diagnostics**.

The Agilent Phantom Human Robot Diagnostics dialog box opens.



2 In the **Profiles** tab, click **Create a new profile**.

3 In the **Create Profile** dialog box that opens, type a name for the profile, and then click **OK**.

4 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

5 *Optional*. If you want the software to prompt the operator for each robot movement, select **Prompt user for each robot move**.

6 Click **Update this profile** to save the changes.

7 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

8 Click **Initialize this profile**.

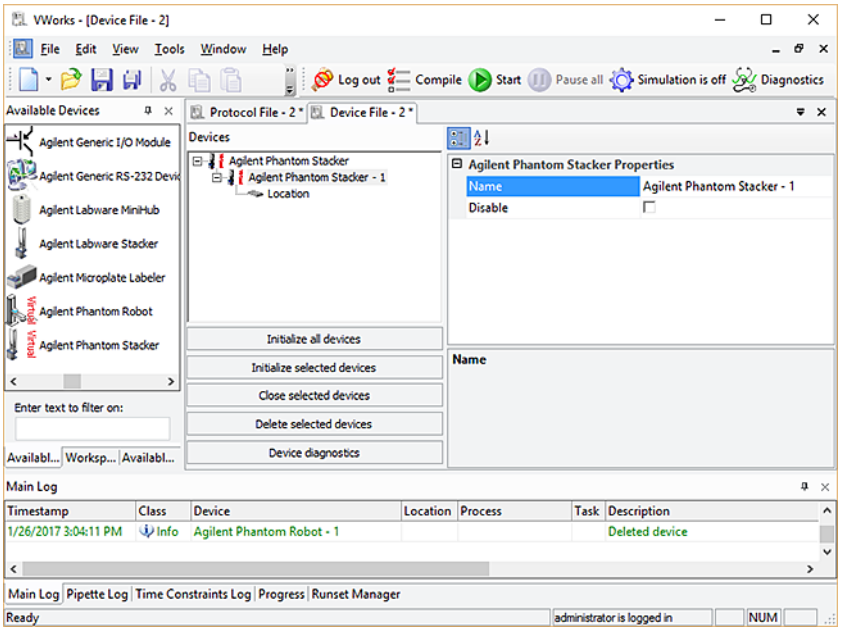
9 Return to the device file and select the profile for the Agilent Phantom Human Robot device.

Agilent Phantom Stacker

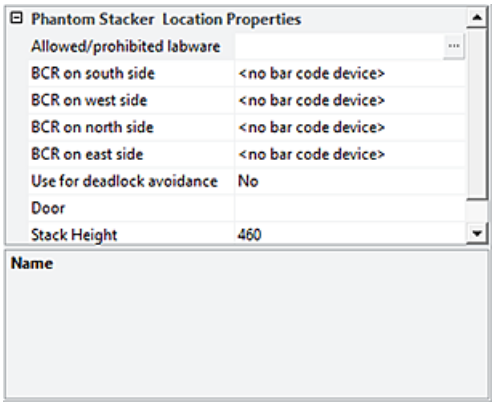
Adding the Agilent Phantom Stacker in the device file

To add the Agilent Phantom Stacker to the device file:

1. In the device file, add the Agilent Phantom Stacker. See "Adding devices" on page 26 for instructions.
2. In the **Agilent Phantom Stacker Properties** area, type the Name of the Phantom Stacker.
3. In the **Devices** area, expand **Agilent Phantom Stacker**, and then select **Location**.



4. In the **Phantom Stacker Location Properties** area, type or set the following:



Property	Description
Allowed/prohibited labware	Permitted labware class for the selected location. Use this field only if you want to limit the labware types. For details on labware classes, see the *VWorks Automation Control Setup Guide*.

Property	Description
BCR on south/west/ north/east side	Not applicable.
Use for deadlock avoidance	Not applicable.
Door	Not applicable.
Stack Height	Type the maximum height (mm) of the stack.

5 Select **File > Save**.

6 *VWorks Plus only*. If an audit trail is being logged, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

Related information

For information about...	See...
Device files	"Adding devices" on page 26
Platepad and Waste Bin devices	"Setting up location devices" on page 32
Other devices	Device user guide
Creating a new protocol	"Workflow for creating a basic protocol" on page 22

Creating a new protocol

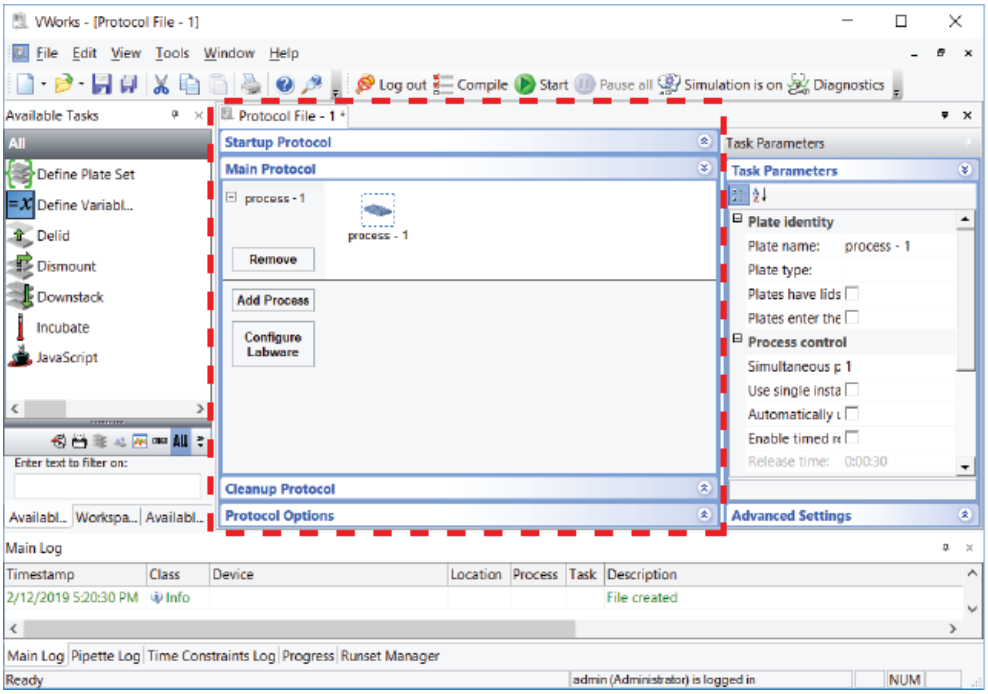
Prerequisites

You must have administrator or technician privileges to create or edit protocols.

Procedure

To create a new protocol:

Select **File > New > Protocol.** A Protocol File tab appears in the VWorks window.



Related information

For information about...	See...
Setting protocol options	"Setting protocol options" on page 42
Adding tasks	"Adding and deleting tasks" on page 67
Setting up startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75
Using advanced protocol features	"Creating a protocol: advanced topics" on page 89

Setting protocol options

When you create a protocol, you need to specify which device file to use with the protocol. Optionally, you can specify additional information to associate with your protocol.

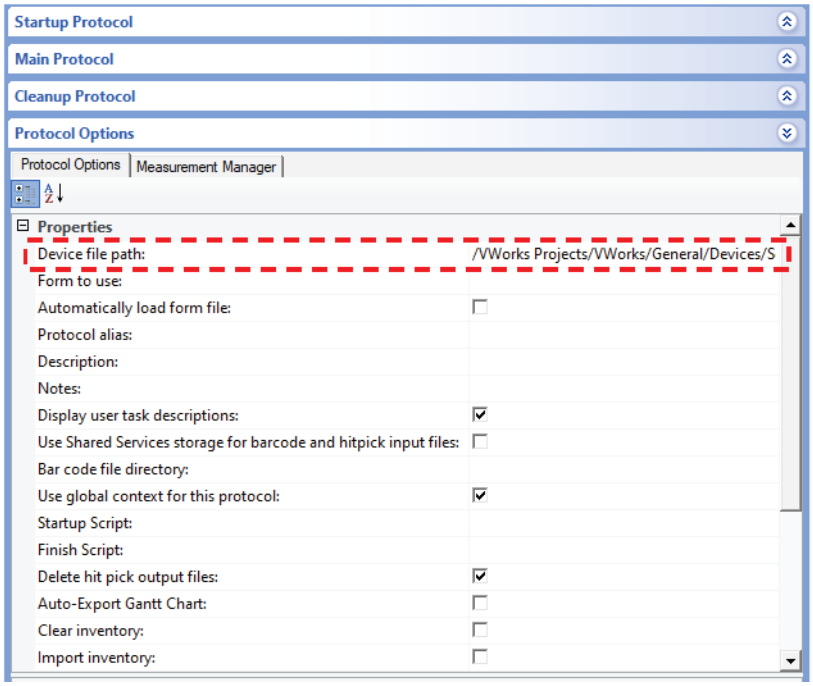
This topic explains the following:

- "Specifying the device file for a protocol" on page 42
- "Adding information about the protocol" on page 43
- "Specifying protocol rules" on page 46

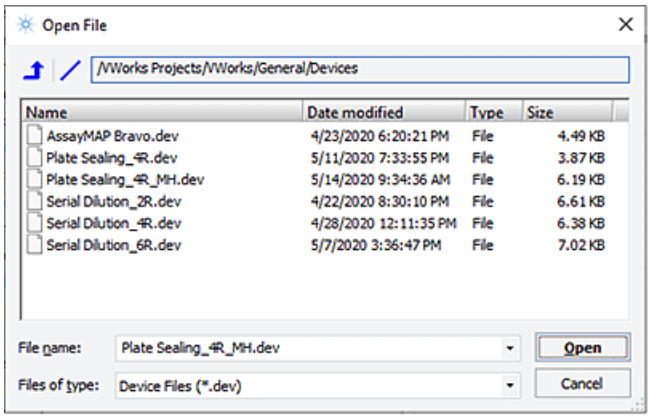
Specifying the device file for a protocol

To specify the device file:

1 In the Protocol area, click **Protocol Options** to view the properties.



2 Click the field next to **Device file path**, and then click the ... button that appears. The Open File dialog box opens.



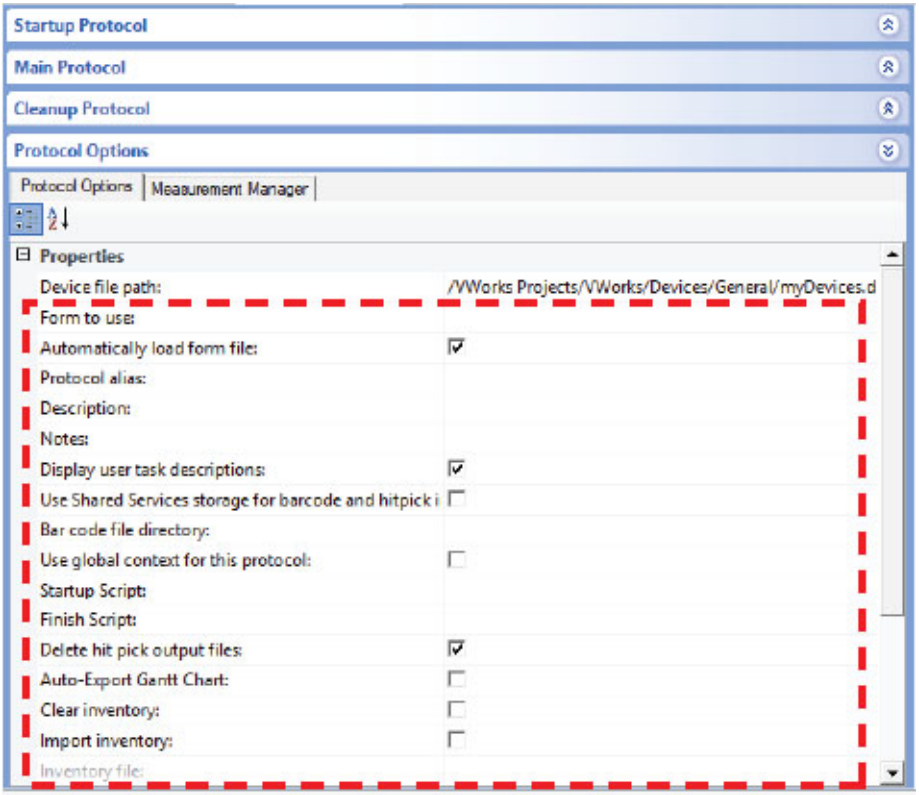
3 Select the correct device file (.dev), and then click **Open.** The path of the device file appears in the Device file path box.

Adding information about the protocol

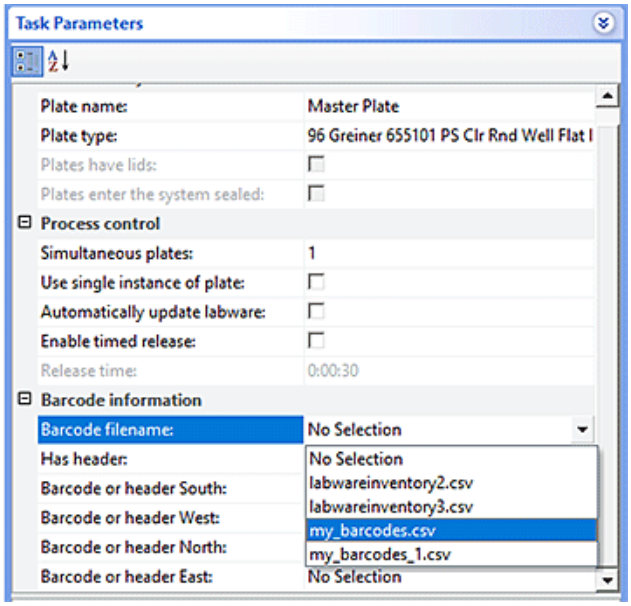
You can add information about your protocol. For example, you can provide a description and some notes about the protocol.

To add information about the protocol:

In the **Protocol Options** area, enter the following information:



Optional protocol information	Description
Form to use	The option that specifies a VWorks form for operators to use as the graphical interface to run a protocol. For details on how to create forms and specify which form to use, see "Creating forms for protocol operators" on page 177.
Automatically load form file	The option that opens the form that is specified in the Form to use field anytime the corresponding protocol is opened.
Protocol alias	Another name for the protocol. The alias is displayed in the software, but the protocol file name is not changed.
Description	A brief description of the protocol.
Notes	Special notes about the protocol.
Display user task descriptions	The option to display the task description (or task label) that you provide instead of the default task label. The label is displayed under the task icon in the protocol.
Use Shared Services storage for barcode and hitpick input files	The option to specify the type of file browser dialog box that opens when selecting barcode and hitpick input files: • Select the check box to open the Shared Services storage file browser dialog box when selecting barcode and hitpick input files. • Clear the check box to open the standard Windows file browser dialog box when selecting barcode and hitpick input files. This file browser dialog box allows you to browse directories in the local file system. *Note:* Selecting this option does not change any previous selections of barcode or hitpick input files.

Optional protocol information	Description
Barcode file directory	The location of the files that contain the barcodes you want to track in the software. The files in this folder will be available in the list that appears in the plate properties area of the protocol, as the following figure shows.  **IMPORTANT** If you select the option **Use Shared Services storage for barcode and hit pick input files**, ensure that you specify a folder in Shared Services storage. Otherwise, the Barcode file directory is in the local file system. To import the barcode and hitpick input files into Shared Services storage, see "Importing and exporting miscellaneous files" on page 49. For instructions on barcode file format, see "Setting plate parameters" on page 59 and "Tracking barcodes" on page 90.
Use global context for this protocol	The option that permits variables to be available across all protocols that also use the global context for variables. **CAUTION** Variables with the same name in other protocols, which also use the global context, will interfere with each other. Always make sure the variable values you want to use globally are applicable to the other protocols that use the global context.
Startup script	The JavaScript code you want to run before the Startup or Main Protocol begins.
Finish script	The JavaScript code you want to run after the Main or Cleanup Protocol finishes.

Optional protocol information	Description
Delete hit pick output files	The option to delete hit-pick output files after the protocol run finishes.
Auto-export Gantt Chart	The option to automatically export the Gantt Chart after the protocol run is finished. The Gantt Chart is exported to Shared Services storage.
Clear inventory	The option to automatically clear the labware information from the inventory database at the start of the protocol run. *Note:* Plate and location group information are retained.
Import inventory	The option to automatically import and use the specified inventory file when starting the protocol run. You specify the inventory file to import in the **Inventory file** box. **IMPORTANT** The list includes only the SQL files that are in OpenLab file storage. To import an SQL file into Shared Services storage, see "Importing and exporting miscellaneous files" on page 49.
Inventory file	*Available only if you selected Import inventory.* The inventory file you want to import and use when starting the protocol run. 1 Click the **Inventory file** box, and then click the browse button that appears. 2 Locate and select the desired inventory (SQL) file. **IMPORTANT** Make sure you import the most current inventory file. Importing an out-of-date inventory file can cause protocol runs to abort.

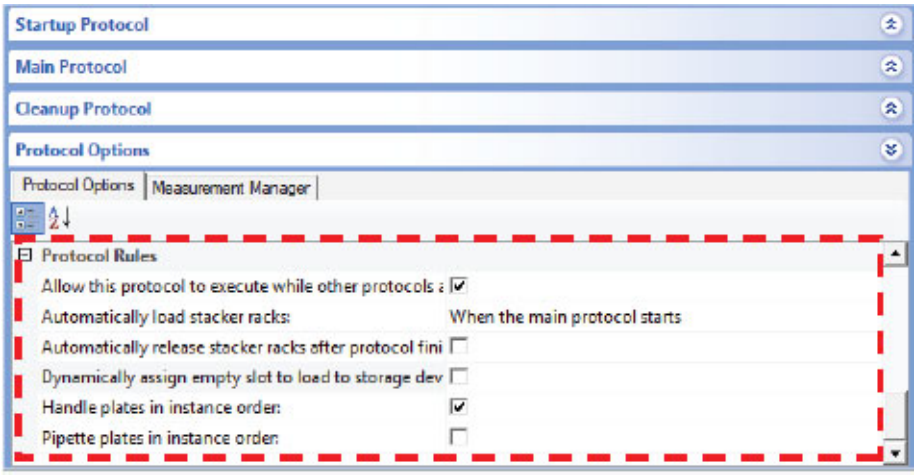
Specifying protocol rules

You can specify certain actions to occur before or after the protocol run.

To specify the protocol rules:

In the **Protocol Options** area, specify the following rules:

- Allow this protocol to execute while other protocols are running
- Automatically load stacker racks
- Automatically release stacker racks after protocol finishes
- Dynamically assign empty slot to load to storage device
- Handle plates in instance order
- Pipette plates in instance order



Allow this protocol to execute while other protocols are running

IMPORTANT

Simultaneously running protocols must specify the same static or configured labware.

This rule allows you to specify the following:

- *Select the check box (default)*. Enables the Runset Manager to run the protocol simultaneously with another protocol.
- *Clear the check box*. Prevents the protocol from running simultaneously with another protocol.

Automatically load stacker racks

Select the **Automatically load stacker racks** rule to require that all racks on stacking devices, such as the BenchCel Microplate Handler, be automatically loaded before either one of the following starts:

- Startup Protocol
- Main Protocol

Automatically release stacker racks after protocol finishes

Select the **Automatically release stacker racks** rule to require that all racks on stacking devices, such as the BenchCel Microplate Handler, be automatically released after the protocol finishes.

Dynamically assign empty slot to load to storage device

Select the rule to have the software assign slots in the storage device according to what is available at that moment. Under this rule, labware can be placed in different locations (not grouped together).

Clear the check box if you want to store labware consecutively in the same carousel or stacker.

Handle plates in instance order

Select the **Handle plates in instance order** rule to require that microplates be processed in the order as they enter the system.

For example:

Two PlateLoc Sealers are used in a system to seal microplates. One of the PlateLoc Sealers runs out of seal material and stops on a microplate.

If this rule is selected, the second sealer would also stop until you load a new roll of seal, start the first sealer, and the microplate at the first sealer is upstacked to its position in the expected order.

If this rule is not selected, the second sealer would continue sealing and upstacking microplates. The microplate at the stopped sealer would be omitted, thus the sequence of microplates will be out of order.

Pipette plates in instance order

Select the **Pipette plate in instance order** rule to require that microplates be processed at pipetting devices in the order as they enter the system.

Select the check box if time-sensitive assays require that each microplate be processed the same way within the same length of time. Doing so ensures the data are comparable across the microplates.

Clear the check box if:

- The protocol has more than one pipette process that uses the same liquid-handling device.
- The duration of one of the pipetting operations is much longer than another.

For example:

A protocol has two processes and both have a pipetting operation that uses the same Bravo Platform.

If the rule is selected, the pipetting operations for one process are completed before the pipetting operations begin for the other process. The first pipetting operation takes significantly longer to complete than the second operation, and the protocol is run several times in succession. The overall time taken for the protocol to complete is much greater than it needs to be because during each cycle the system had to wait for the slower pipetting operations to complete for all the microplates in the process before it could continue.

If the rule is not selected, a microplate from the fast pipetting process can be delivered to the Bravo Platform after a microplate from the slow pipetting process, followed by another microplate from the slow pipette process, and so on. This reduces the bottleneck at the Bravo Platform because it allows the faster process to continue, and its second cycle in the series to start before the first cycle is complete.

Related information

For information about...	See...
Specifying a device file for the protocol	"Specifying the device file for a protocol" on page 42
Importing files created in other software, such as barcode and JavaScript files	"Importing and exporting miscellaneous files" on page 49

Importing and exporting miscellaneous files

This topic describes how to import and export miscellaneous files. VWorks technician privileges or greater are required to import miscellaneous files into the Shared Services storage.

For instructions on how to export and import protocols and associated records, such as forms, device files, and labware, see "Exporting and importing protocols and associated records" on page 703.

Miscellaneous file types

The following table lists files that are created externally in other software. Some of these miscellaneous files must be imported into Shared Services storage before they are available for use in the VWorks software. For example, if your protocol references a JavaScript (.js) file, you import the file before you can reference it in the protocol.

VWorks Plus only. The following files may be treated as records of interest to enable compliance with Part 11 of Title 21 of the Code of Federal Regulations (21 CFR Part 11).

File type (file extension)	Description
AssayMAP Method files (.mth)	Method files created in Protein Sample Prep Workbench 4.0 are stored in Shared Services storage. To share a method on another computer, the method file must be exported and then imported on the other computer. The default AssayMAP methods are installed in the following location: /VWorks Projects/VWorks/AM Methods/ The AM Methods folder contains a folder for each app and utility. The file names for the default method files (.mth) are the same for multiple apps and utilities. However, only the app or utility used to create a method can open that method. To help prevent importing a method file into the wrong subfolder, the software prepends the folder name to the file name when you export a method. For example, if you export the 5uL Cartridge Default Settings.mth from the AM Affinity Purification 3.0 folder, the file name becomes AM Affinity Purification 3.0_5uL Cartridge Default Settings.mth No overwrite is allowed for this file type.

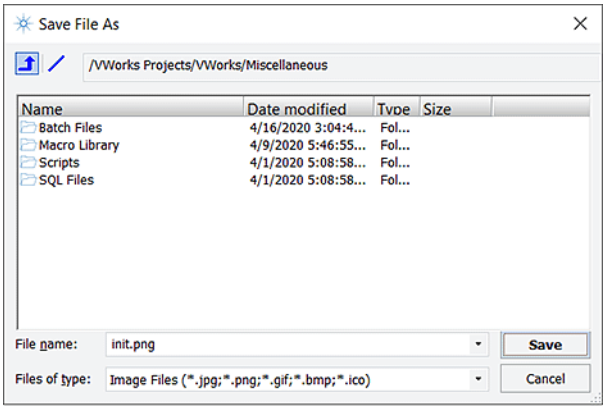
File type (file extension)	Description
Barcode files (.bar, .csv, or .txt)	Barcode files may be stored in either of the following locations: • Shared Services storage. Use the procedure in this topic to import barcode files into Shared Services storage. • Local file system. No importing is required. For file format requirements, see "Setting plate parameters" on page 59 and "Tracking barcodes" on page 90.
Excel files (.xls, .xlsx, .xlsm)	Import any Excel files that you want VWorks to use.
Gantt chart files (.gnt)	Import Gantt chart files that you want to open and view in the VWorks window.
Hitpick input files (.csv)	Hit-picking and plate replication operations require an input file that contains information about the source microplate. Hitpick input files may be stored in either of the following locations: • Shared Services storage. Use the procedure in this topic to import hitpick input files into Shared Services storage. • Local file system. No importing is required. For file format requirements, see "Hit Pick Replication (Agilent Bravo)" on page 526.
Image and icon files (.jpg, .png, .gif, .bmp, or .ico)	Import any image and icon files that you want to use for the following: • Labware definitions • VWorks forms
Inventory files and Experiments database files (.sql)	Import the .sql files that you want VWorks to use. For details about setting up the Inventory, Experiments, and E-Signatures databases, see the *VWorks Automation Control Setup Guide*.
JavaScript files (.js)	Import the JavaScript files that you want VWorks to use. For details on using JavaScript, see "Creating a protocol: advanced topics" on page 89.
PDF files (.pdf)	Import the .pdf files that you want VWorks to use.

Importing miscellaneous files

To import miscellaneous files into Shared Services storage:

1 In the VWorks window, click **File > Import Misc Files**. The Select a Miscellaneous File for Import dialog box opens.

2 Select the file type, select the file, and then click **Import**. The Save File As dialog box opens.



3 Select the storage folder, and then click **Save**.

If a file of the same name already exists, one of the following messages appears when you try to import it:

- If you are a VWorks technician, the message states that this operation can only be performed by VWorks Administrators.
- If you are a VWorks administrator, the message informs you that the file already exists. In this case, click Yes to proceed, or click No to cancel.
- *AssayMAP Method files only*. A message informs you that no overwrite is allowed. In this case, you must rename the file to continue importing it.

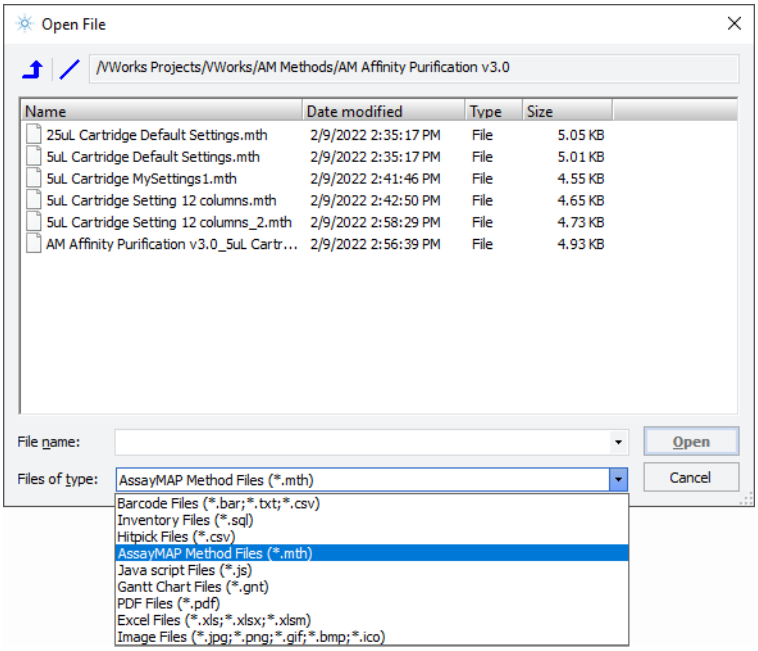
4 *VWorks Plus only*. The Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

Note: The imported miscellaneous files are exempt from changes in record state. An audit trail is always logged for exempt state records.

Exporting miscellaneous files

To export miscellaneous files from Shared Services storage:

1 In the VWorks window, click **File > Export Misc File**. The Open File dialog box appears.



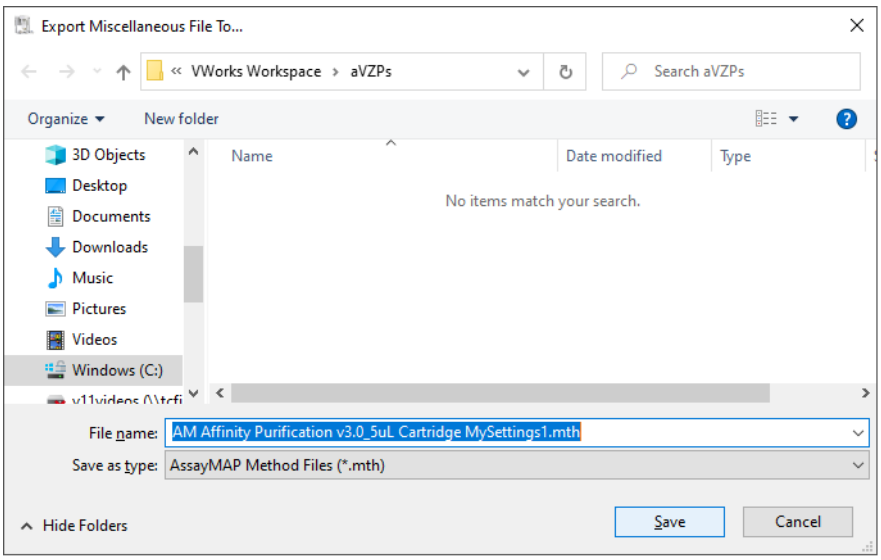
2 In the **Open File** dialog box:

a Click the file type in the **Files of type** list.

b Use the button to navigate to the folder in Shared Services storage.

c Select the file to be exported, and then click **Open**.

The **Export Miscellaneous File To** dialog box opens.



3 In the **Export Miscellaneous File To** dialog box, ensure that the correct **Save as type** is selected, navigate to the export location, and then click **Save**.

IMPORTANT

For AssayMAP Method files, Agilent recommends retaining the folder name as part of the exported file name to help ensure that the file is imported to the correct method subfolder for the corresponding AssayMAP app or utility.

Related information

For information about...	See...
File storage	"Architecture and features overview" on page 2
Adding other information about the protocol	"Setting protocol options" on page 42
Labware Editor description	*VWorks Automation Control Setup Guide*
VWorks forms	"Creating forms for protocol operators" on page 177.
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Configuring labware

About this topic

After you create a new protocol, you can configure labware for devices that have locations for labware placement. When labware is configured at a device location, the labware might be moved from its location for deadlock avoidance or be used as a counterweight, but returned to the same location by the end of the run.

For example, you can configure labware that will be stored at Bravo deck location 2. During the protocol, the labware will be moved to deck location 5 temporarily for processing, and then returned to deck location 2.

This topic explains how to configure labware. For more information about configured labware and how it is used in a protocol, see "Planning labware use" on page 24.

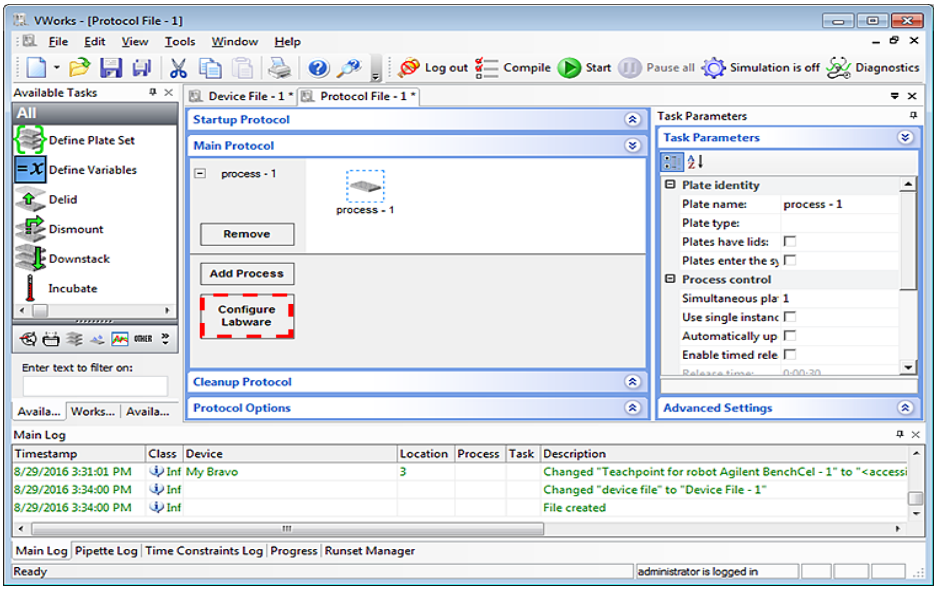
Configuring labware for Auto managed counterweight

If you are configuring labware for use as an Auto managed counterweight, see "Example: Auto managed counterweight mode" on page 356 for instructions on how to configure labware for this feature.

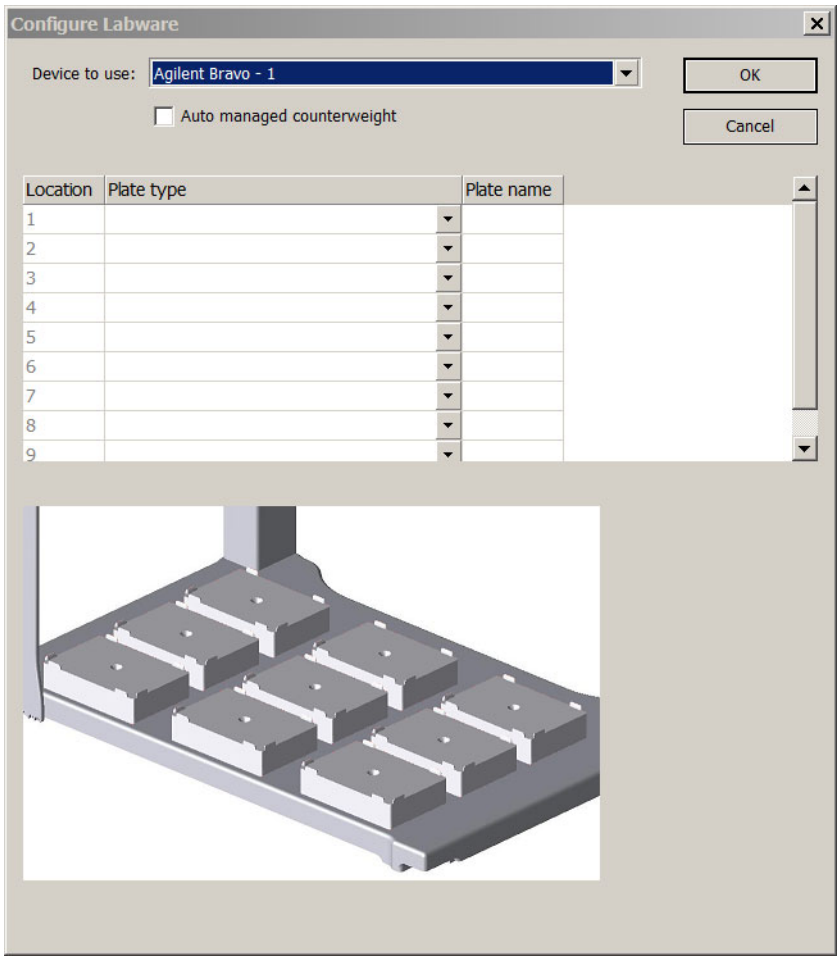
Procedure to configure labware

To configure labware:

1 In the **Main Protocol** of the VWorks window, in the area below the protocol process, click **Configure Labware.**



The Configure Labware dialog box opens.



2 In the **Device to use** list, select the device on which you want to configure labware.

Note: Devices that are in the device file and have labware placement locations appear in the list. Exceptions include storage devices such as the Labware MiniHub, centrifuge devices, and stacking devices.

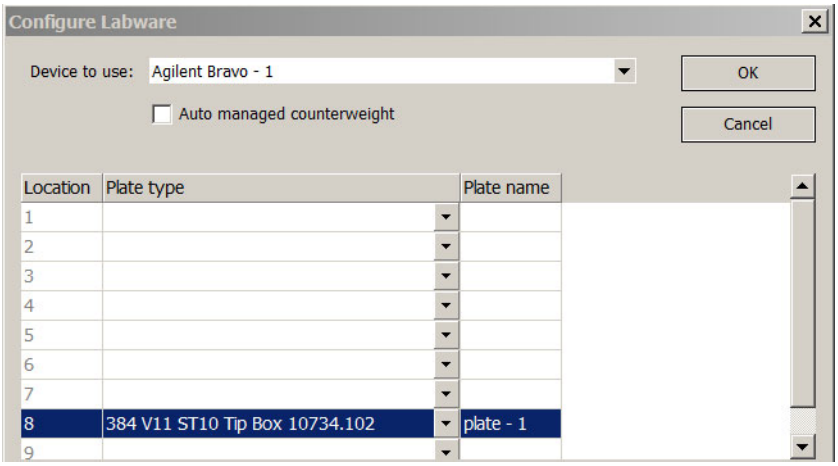
3 If you are configuring labware as an auto-managed counterweight for use with the Centrifuge or Centrifuge with Loader, select **Auto managed counterweight.** See "Example: Auto managed counterweight mode" on page 356 for further information and instructions on using this feature.

4 In the **Location** table:

a In the **Plate type** column, select the labware for the corresponding location.

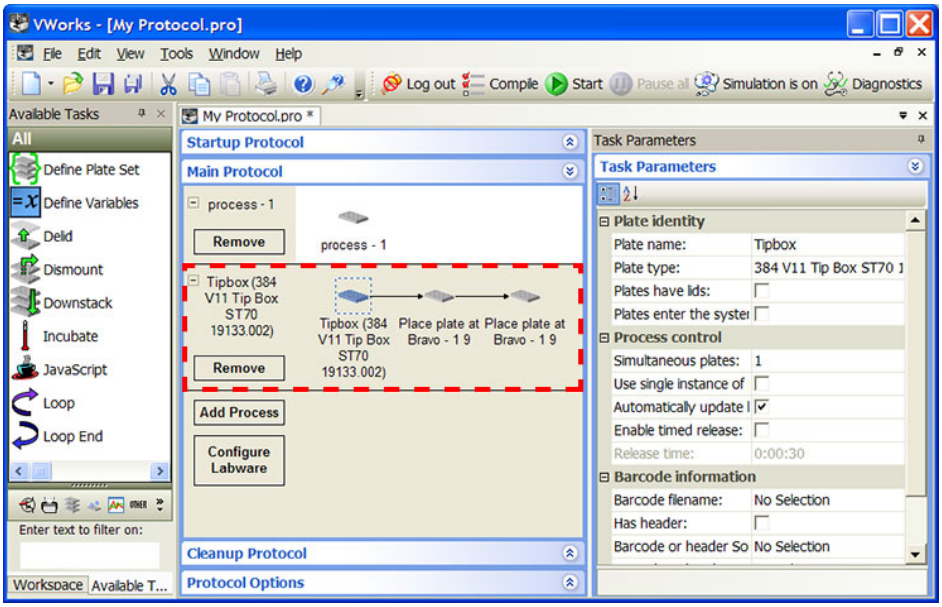
b In the **Plate name** column, double-click the text box, and then type a name for the labware you selected.

Note: For the Bravo Platform, configured static labware (from protocols created in VWorks4 version 6.2.3 or earlier) will appear in the table and graphic. However, configured accessories do not appear in the table and graphic.



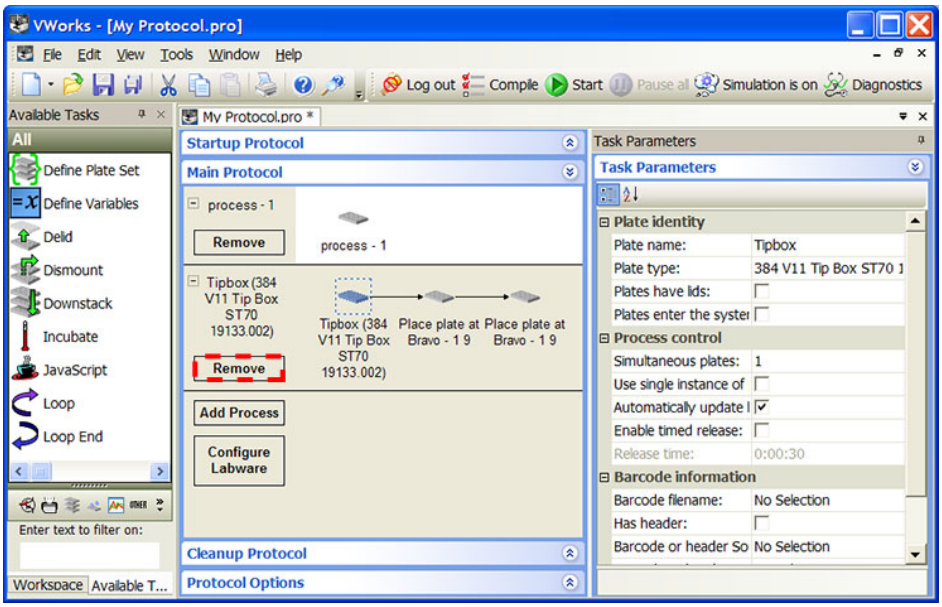
5 When you are finished, click **OK** to save the information and close the Configure Labware dialog box.

In the Main Protocol area, the labware appears. Click + to expand and view the labware icon. Notice that two Place Plate tasks are automatically added, one to indicate its starting location and the other to make sure it returns to its starting location at the end of the protocol.



6 Select the configured labware icon. In the **Task Parameters** area, set the plate properties. For a description of the plate properties, see "Setting plate parameters" on page 59.

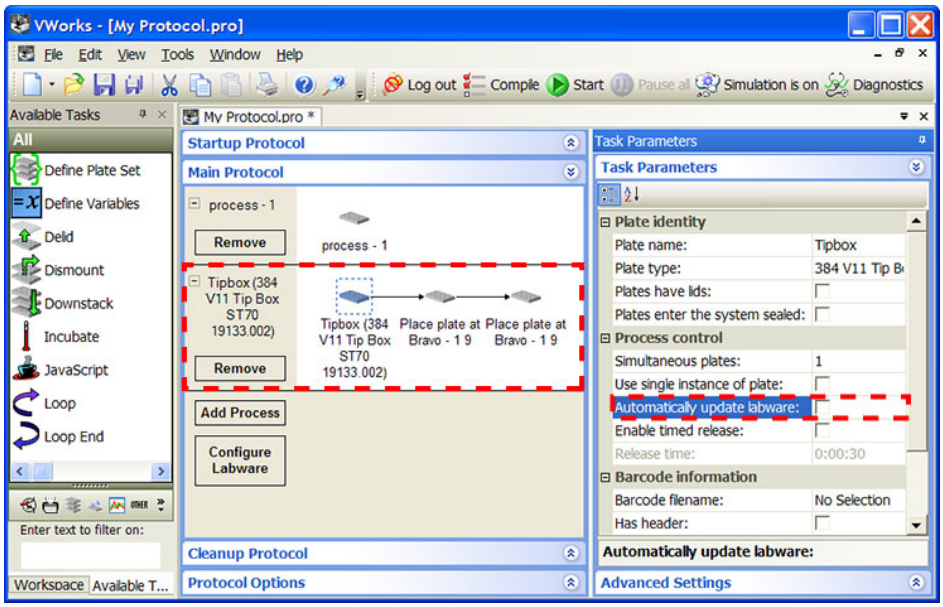
7 If you want to remove the configured labware, click **Remove.**



8 To convert a configured labware to a process plate (labware that is transferred into the system and out of the system during the run), in the **Process Control** area, clear the **Automatically update labware** check box.

The background in the configured labware process area becomes white, indicating that it is converted to a protocol process.

Note: You can use the **Automatically update labware** option to update a labware location throughout the protocol. For example, clear the check box, and then update a configured labware location from Bravo deck location 3 to location 5. Then select **Automatically update labware.** The software automatically makes the changes throughout the protocol.



Using configured labware

When you add a task in the process, you can use available configured labware by selecting it from the **Location, plate** list. In addition, if more than one subprocess uses the same configured labware, and the subprocesses are in different protocol processes, you can specify the sequence in which the subprocesses will be performed. For instructions, see "SubProcess (Agilent Bravo)" on page 450.

Related information

For information about...	See...
Planning labware use	"Planning labware use" on page 24
Process plates	"About protocols, processes, and tasks" on page 18
Planning for devices and accessories in the protocol	"Planning device and accessory use" on page 24
Static labware	"Configure Static Labware" on page 361

Adding processes

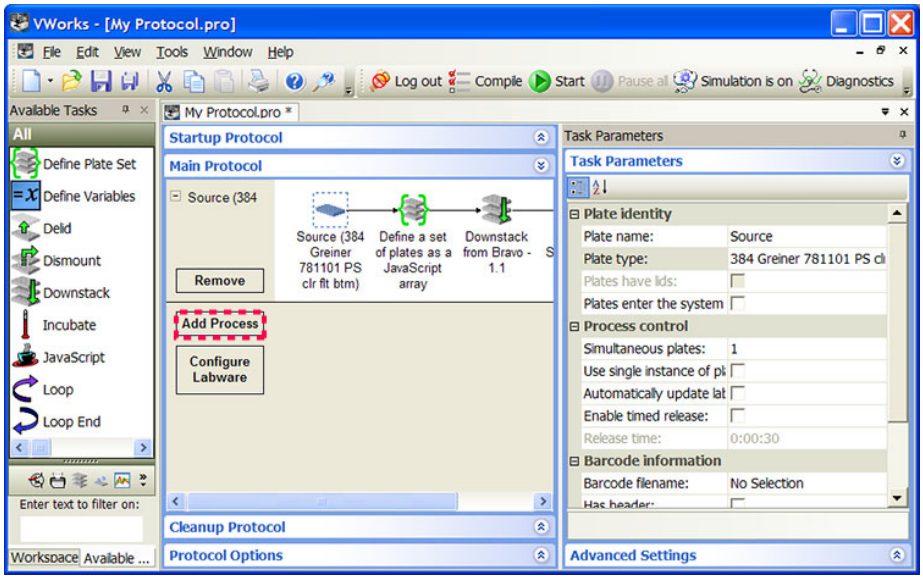
About processes

You can add processes at any time when you are creating the protocol. In addition, you can add multiple processes in one protocol.

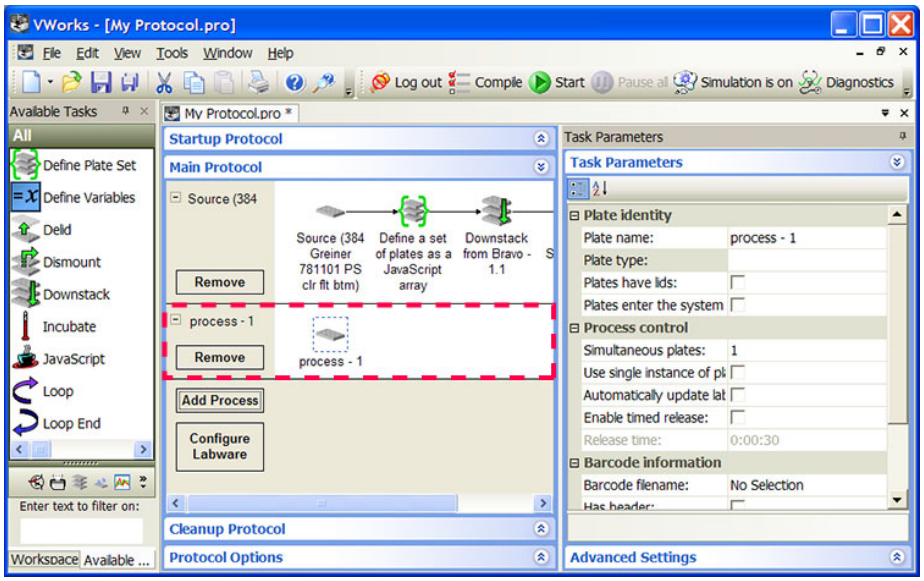
Procedure

To add a new process:

In the **Main Protocol,** click **Add Process.**



A new process lane appears.



Related information

For information about...	See...
Processes	"About protocols, processes, and tasks" on page 18
Process plates	"About protocols, processes, and tasks" on page 18
Configured labware	"Configuring labware" on page 53

Setting plate parameters

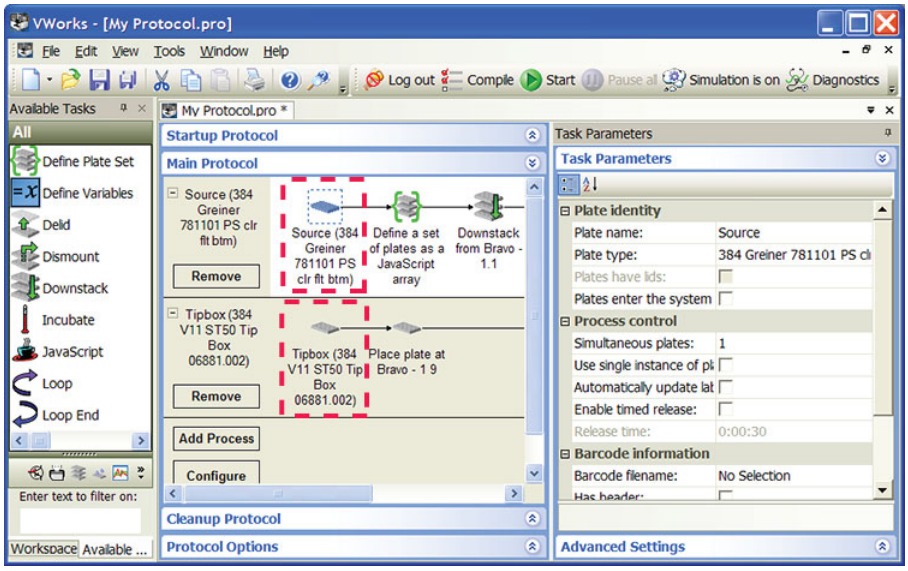
About this topic

You need to set parameters for process plates and configured labware. This topic explains how to set the parameters for both.

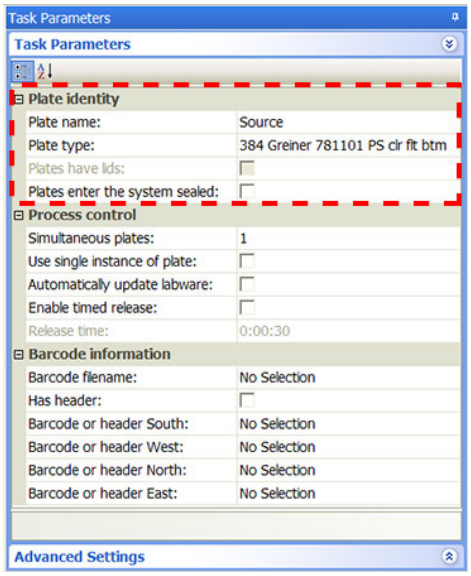
Procedure for setting plate parameters

To set plate parameters:

1 In the **Protocol** area, select the process plate icon or the configured labware icon.

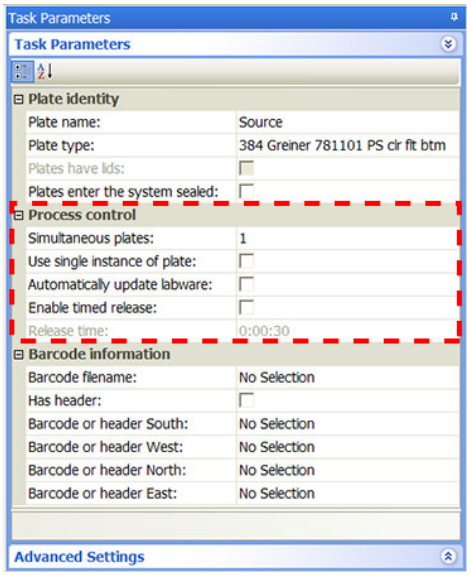


2 In the **Plate identity** area, set the following parameters:



Property	Description
Plate name	The name that will help you identify the labware. For example, you can use Source Plate or Destination Plate.
Plate type	The type of labware. The list of labware types is created in the Labware Editor.
Plates have lids	The indicator that the labware entering the system has a lid. Select the check box if the labware has a lid. Clear the check box if the labware does not have a lid. *Note:* The selection is only available if the **Can have lid** option is selected in the Labware Editor.
Plates enter the system sealed	The indicator that the labware entering the system is sealed. Select the check box if the labware is sealed. Clear the check box if the labware is not sealed. *Note:* The selection is only available if the **Can be sealed** option is selected in the Labware Editor.

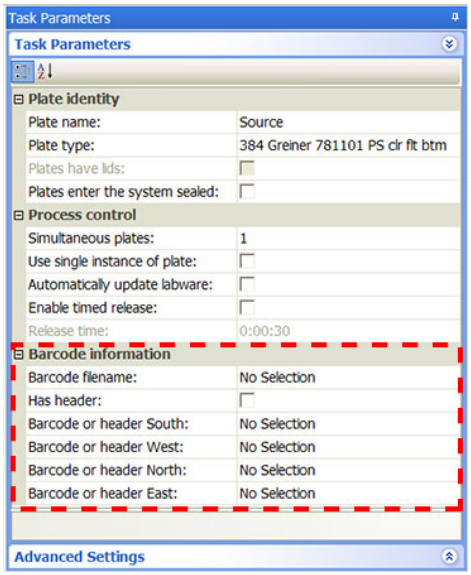
3 In the **Process control** area, set the following parameters:



Property	Description
Simultaneous plates	The maximum number of labware instances that are processed in the system at one time. For details about how to determine the optimum number, see "Determining the number of simultaneous plates" on page 63.
Use single instance of plate	The indicator that the plate type has only one instance and will be used repeatedly during the run. For example, you might have a single source plate from which you will aspirate repeatedly during a protocol. Select the check box if the labware has only one instance. Clear the check box if the labware has more than one instance.
Automatically update labware	The indicator that the labware is configured labware. Select the check box if the labware is a configured labware and will be used by one or more tasks in the protocol process. When you revise the protocol, the changes are automatically reflected in the configured labware process. Clear the check box if you want to convert the configured labware to a process plate.
Enable timed release	The indicator that the labware is delivered into the system in timed intervals. If you select this property, you also need to set the Release time. Select the check box if you want the system to deliver the labware in timed intervals. Clear the check box if the system can deliver the labware without waiting the specified time.

Property	Description
Release time (Sec)	The length of time to wait before delivering the next labware instance into the system. For details about how to determine the release time, see "Determining the correct microplate release rate" on page 64.

4 In the **Barcode information** area, set the following parameters:



Property	Description
Barcode file name	The file you want to use to verify barcodes on incoming labware. Select from the list of barcode files. Depending on the setting in Protocol Options, the list of barcode files is from Shared Services storage or from the local file system. If you do not see a list, make sure you specified the file location in the Protocol Options. See "Adding information about the protocol" on page 43. For details on the file structure and content, see "Input files" on page 90. To import barcode files into Shared Services storage, see "Importing and exporting miscellaneous files" on page 49.
Has header	The indicator that the first line of the barcode file contains a header. Select if the code file name contains a header. Clear the check box if the file does not contain a header.

Property	Description
Barcode or header South/West/North/ East	The column ID in the barcode file. Select the column ID for the side of the labware. For example, if the barcode is on the east side of the labware, select the column ID in the **Barcode or header East** field. Select from the list of column IDs. If you do not see a list, make sure you specified the file location in the Protocol Options and selected the file from the **Barcode file name** property.

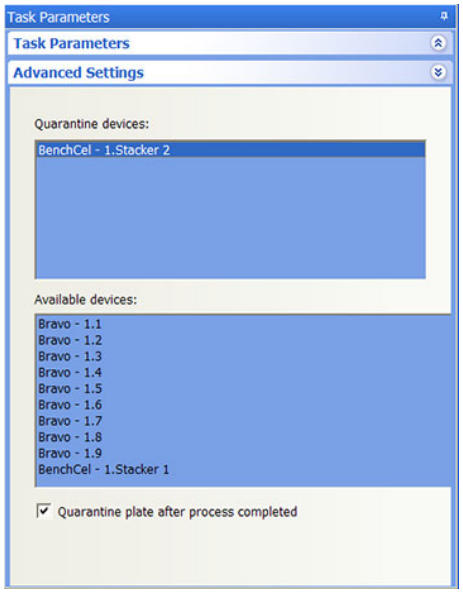
5 *Optional*. Reserve a device to store quarantined labware. Labware can be quarantined (moved aside and not processed) if a barcode label is misread, the labware orientation is incorrect, or the wrong labware type is detected. The quarantine option allows the system to continue running the protocol even though it is unable to resolve problems with the labware.

To reserve a device to store quarantined labware:

a Click **Advanced Settings.**

b In the **Available devices** list, double-click the device you want to use to store quarantined labware. The device appears in the **Quarantined devices** list.

c Select **Quarantine plate after process completed** if you want to quarantine labware after the protocol is finished. Clear the check box if you want to quarantine the labware as soon as it is encountered in the run.



For more information about how to set up quarantine criteria, see "Setting up automated error responses" on page 735.

Determining the number of simultaneous plates

Factors to consider

The number of simultaneous plates you specify depends on the following:

- The number of positions available during a protocol process.

In general, you can specify one simultaneous plate for every task in the protocol, because each task typically uses one microplate position. For example, if your protocol downstacks a microplate, seals the microplate, labels the microplate, and then upstacks the microplate, you have three positions available: one in the robot grippers, one on the microplate sealer, and one on the microplate labeler.

- The number of positions in the system that will incubate microplates.

  A Labware MiniHub that has 3 shelves has 12 possible microplate positions. If your protocol downstacks a plate, dispenses liquid, incubates the microplate in the 12-position MiniHub, and then dispenses more of the same liquid, you have 14 positions available: one in the robot grippers, one on the dispenser, and 12 in the MiniHub.

- The number of microplates in a Reorder task.

  If the protocol includes a Reorder task, the number of simultaneous plates must be equal to, or greater than, the number of microplates in the Reorder task.

- The slowest or rate-limiting task.

  A task such as a long read step or wash task on a single device can impact the number of microplates that can enter the system.

How throughput is impacted

The number of simultaneous plates you specify can impact throughput as follows:

- *The value is too high*. The protocol run might slow down because the robot will move around to avoid a deadlock. (A deadlock occurs when the number of locations available in the system is less than the number of microplates in the system, and the protocol stops.)
- *The value is too low*. The time for the protocol run can be longer than desired.

Determining an optimum number of simultaneous plates

By default, the number of simultaneous plates is set to one for each protocol process. If your protocol process uses multiple devices, you can increase the throughput of the system by increasing the number of plate instances to be processed simultaneously. The optimum number should balance high throughput and deadlock avoidance.

To determine the optimum number of simultaneous plates:

1. Run the protocol in simulation, noting the protocol process time in the log.
2. Increase the number of simultaneous plates.
3. Repeat steps 1 and 2 until the simulated process time no longer decrease. The simultaneous plates value is optimum when the process time no longer decreases.

IMPORTANT

If “Attempting to avoid deadlock by...” messages appear in the log, the protocol might have too many simultaneous plates. Decrease the number of simultaneous plates to decrease the likelihood of a deadlock during the protocol run.

Determining the correct microplate release rate

You can use the **Enable timed release** and **Release time** plate properties to control the microplate release rate and prevent bottlenecks and deadlocks. An optimum release rate should balance the number of microplates entering into the system with the number of microplates exiting the system.

Bottlenecks can occur when tasks or incubation times cause the number of incoming microplates to be greater than the number of outgoing microplates. To prevent the bottleneck, you can limit the rate of microplates entering into the system.

IMPORTANT

Make sure you determine the optimum number of simultaneous plates before you change the microplate release rate.

To determine the optimum microplate release rate:

1. Run the protocol using empty labware.
2. During or after the run, open the Gantt Chart dialog box to find the task that is causing the bottleneck and determine the length of time for the task to finish. (For information about the Gantt Chart dialog box, see "Tracking the run progress of instances or devices" on page 330.)
3. Select **Enable timed release.** Type the length of the bottlenecking task time in the **Release time** box.
4. Run the protocol again using empty labware.
5. Determine whether the bottleneck is still occurring.
   - If the bottleneck is resolved, determine if previous tasks can process microplates during the wait time so that you can improve throughput. If so, decrease the release time.
   - If the bottleneck still occurs, increase the release time.
6. Repeat steps 1 through 5 until the bottleneck is resolved and the throughput is optimized.

For example:

A protocol is running with three simultaneous plates. A pipetting task in the middle of the protocol takes 3 minutes. During this time, two other microplates have already entered the system and must wait for the pipetting task to finish. A deadlock error occurs, because the system is unable to find a storage location for the third microplate that entered the system.

By turning on the timed release property and setting the release time to 180 seconds (or 3 minutes), the system will wait 3 minutes after the first microplate is in the system before delivering the next microplate. However, a microplate-piercing task and a microplate-shaking task take a total of 1 minute before the pipetting task. So the release time can be decreased to 120 seconds (or 2 minutes) to improve throughput.

Note: Bottlenecks in a protocol might not always cause a deadlock error. Instead, microplates might wait on platepads. Monitor the dry protocol run to check for non-error-causing bottlenecks.

Related information

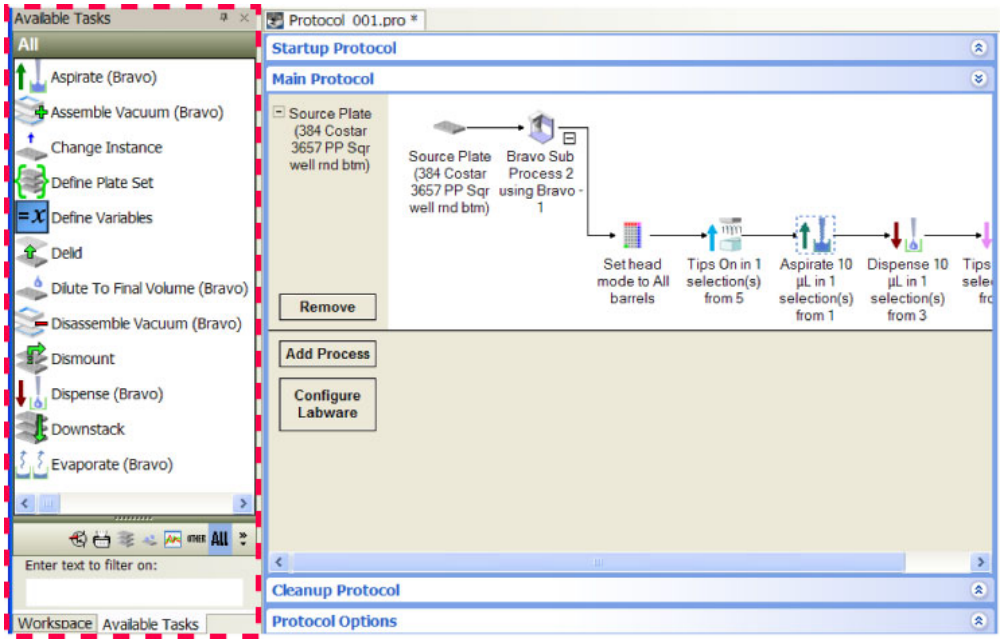
For information about...	See...
Labware Editor	*VWorks Automation Control Setup Guide*
Barcode file folder location	"Setting protocol options" on page 42
Monitoring run progress of instances, processes, and devices	"Tracking the run progress of instances or devices" on page 330

For information about...	See...
Setting up quarantine criteria	"Setting up automated error responses" on page 735
Quarantining labware that are downstacked from a BenchCel stacker	"Downstack" on page 414

Adding and deleting tasks

Viewing the list of available tasks

The list of available tasks you can use in the protocol are in the Available Tasks area.



The tasks that are available depend on a number of factors:

- *Devices*. Tasks are associated with devices. Only the tasks associated with the devices in the current device file are displayed.
- *Startup or Cleanup protocol*. Some tasks appear only in the Startup or Cleanup protocols. For a description of Startup and Cleanup protocols, see "Setting up Startup and Cleanup Protocol processes" on page 75.
- *Subprocess*. Some tasks are grouped in a device subprocess, so you can view the tasks only when you select the subprocess.

Filtering the list of available tasks

You can filter and display a subset of the task icons using one of the following methods:

- Click a task filter button.
- Type filtering text.

Using the task filter buttons

To use the filter buttons:

Click one of the following filter buttons below the list of available tasks:



Item	Task display filter	Description
1	Not used	
2	Plate Handling	Tasks that move labware or might change the labware characteristics without changing its contents. For example, Centrifuge, delid, and Seal are plate-handling tasks.
3	Plate Storage	Tasks that move labware into or out of storage devices such as the Labware MiniHub. For example, Downstack and Reorder are plate storage tasks.
4	Liquid Handling	Tasks that perform liquid-handling operations. For example, Aspirate and Dispense are liquid handling tasks.
5	Reading	Tasks that perform microplate-scanning operations. For example, the Read task is a microplate-scanning task using the Perkin Elmer Fusion Reader.
6	Other	Displays scheduling tasks, such as Define Variable, Loop, and Spawn Process.
7	All	Displays all the available tasks.

Typing filtering text

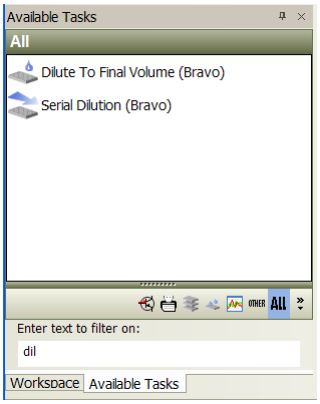
To type filtering text:

1 Click one of the filter buttons. See "Using the task filter buttons" on page 67.

For example, if you want to search all the tasks, click **All.** However, if you only want to search the liquid-handling tasks, click the **Liquid Handling** filter button.

2 Type text you want to use to filter the task list in the **Enter text to filter on** box below the list of available tasks. As you type each character, the Available Tasks list changes to meet the filter requirement.

For example, if you type `dil`, only the dilution tasks remain in the Available Tasks list.



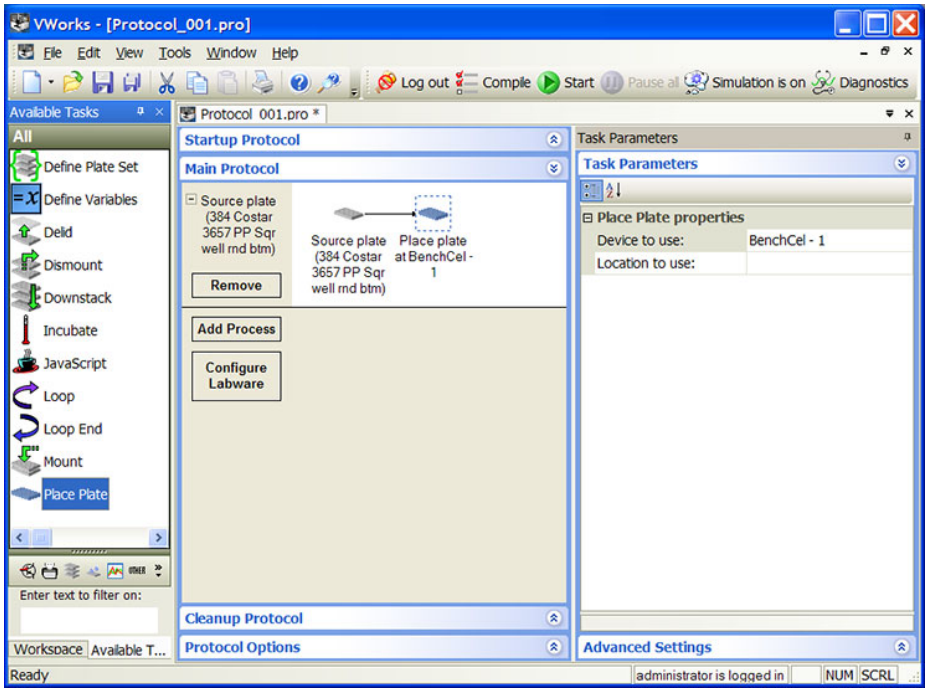
Adding tasks in a protocol

For details on how to use macros to insert a series of tasks, see "Using macros to create protocols" on page 265.

To add a task in a protocol:

1 In the **Available Tasks** area, double-click a task. Alternatively, drag a task from the **Available Tasks** area into the Protocol area. The task icon appears in the Protocol area.

In the following example, the Place Plate task is added to the protocol.



2 Set the task parameters. For a description of the task parameter, see one of the following:

- "Setting parameters for microplate-handling tasks" on page 343
- "Setting parameters for microplate storage tasks" on page 411
- "Setting parameters for liquid-handling tasks" on page 449
- "Setting parameters for scheduling tasks" on page 639

Deleting tasks from a protocol

To delete tasks from a protocol:

1 In the **Protocol** area, select a task.

2 Press DELETE**.** A confirmation message dialog box opens.

3 Click **Yes** to confirm the delete action. The task icon is removed from the protocol.

IMPORTANT

Be aware of dependencies between tasks before you delete them. For example, the Tips On (Bravo) task relies on the pipette channel specifications in the Set Head Mode (Bravo) task to determine the number and position of pipette tips to install.

Related information

For information about...	See...
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639
Using a macro to add a series of tasks	"Using macros to create protocols" on page 265

Specifying time constraints between dependent tasks

About dependent tasks and time constraints

After you add tasks to a protocol, you can specify a time interval between two dependent tasks. For example, you can specify that a microplate must be read within 30 minutes after a stop solution is added.

You can specify time constraints between any two non-scheduling tasks in the Startup, Main, or Cleanup Protocol. In addition:

- Both time-dependent tasks can be within a subprocess or a loop. If a Change Instance task is used in a loop, the time limit is used for every plate instance.
- One of the tasks can be within a subprocess and the other outside of the subprocess.
- One of the tasks can be within a loop and the other outside of the loop. If the constraining task is before a loop and the constrained task is within the loop, the constraint applies to the first loop only. If the constraining task is within the loop and the constrained task is after the loop, the constraint applies to the last loop only.
- Any number of tasks can be added between the two dependent tasks. However, if the time it takes to complete the enclosed tasks exceeds the time constraint of the dependent tasks, a compiler error results.

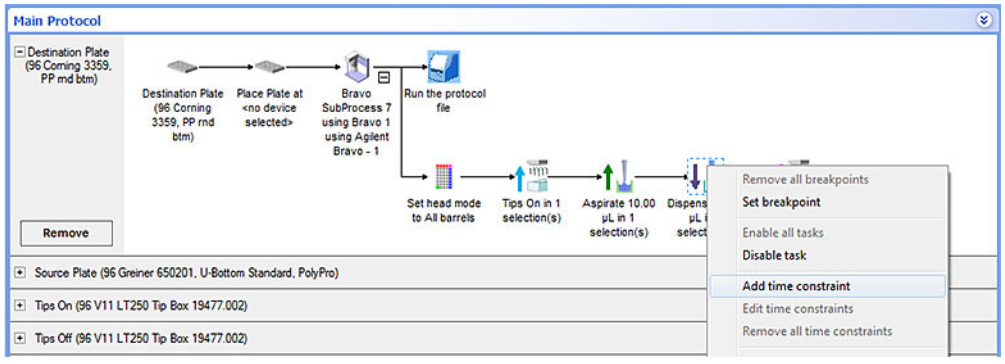
During a protocol run, if the time limit is exceeded, the run continues, but a message appears in the Time Constraint Log to alert you of this occurrence. If the task is finished before the time limit, the software waits until the minimum time limit is reached before proceeding to the next task.

Adding a time constraint

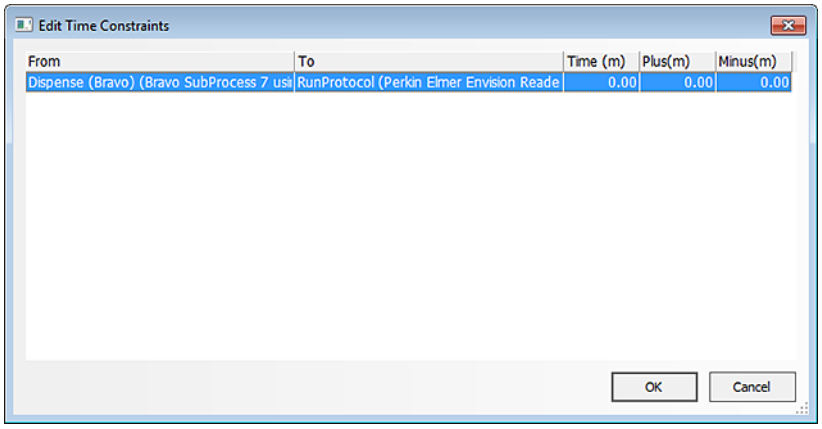
To set a time constraint between two dependent tasks:

1 In the protocol, CTRL+click the two dependent tasks to select both tasks.

2 Right-click one of the two tasks, and then select **Add time constraint.**



The Edit Time Constraint dialog box opens. Notice that the task that runs first is the constraining task, and the second task is the constrained task.



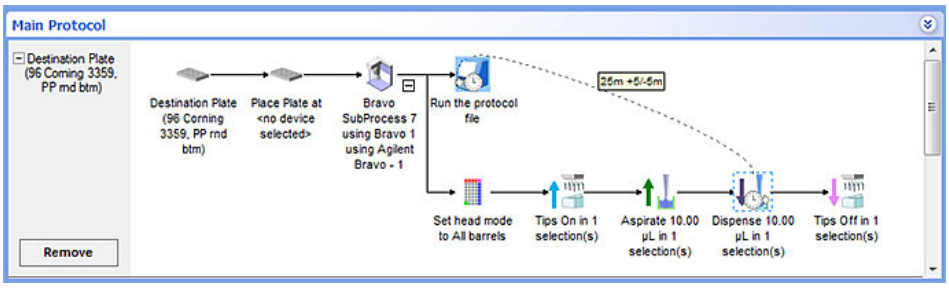
3 Specify the length of time between the two dependent tasks.

Parameter	Description
From	The first of the two dependent tasks you selected. You cannot edit this field.
To	The second of the two dependent tasks you selected. You cannot edit this field.
Time (m)	The length of time, in minutes, that is permitted for the first task. The minimum value is 1 minute.
Plus (m)	The time tolerance upper limit. For example, if you specified a total time of 25 minutes +5 minutes, the maximum time permitted for the first task is 30 minutes. The software will proceed to the next task after allowing 30 minutes for the first task to complete. The minimum tolerance is 0.5 minute.
Minus (m)	The time tolerance lower limit. For example, if you specified a total time of 25 minutes - 5 minutes, the minimum time permitted for the first task is 20 minutes. The software will wait 20 minutes before proceeding to the next task even if the first task is completed in less time. The minimum tolerance is 0.5 minute.

4 Click **OK** to save the changes and return to the protocol.

A clock appears on each of the two task icons to indicate that they are time-dependent. An arrow pointing away from the clock indicates that the task sets a constraint. (In the following example, the Dispense task is the constraining task.) An arrow pointing into the clock indicates that the task is constrained by an earlier task. (In the example, the Run the protocol file task is the constrained task.)

When you rest the cursor on either task icon, a dotted line and the specified time interval appear.



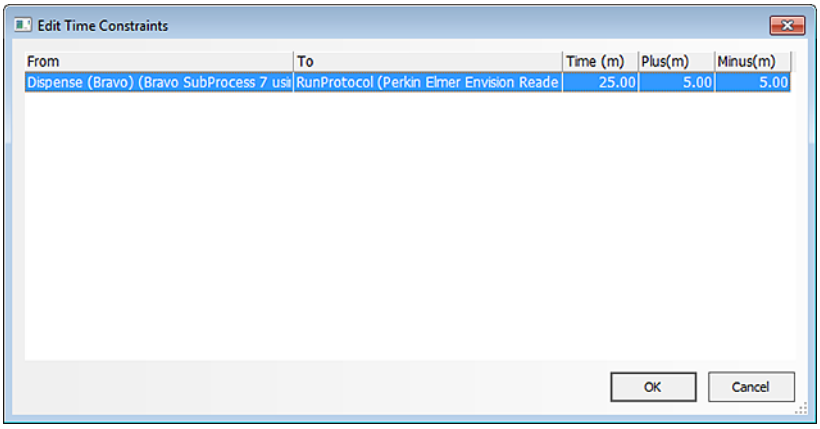
Note: A task can both constrain a later task and be constrained by an earlier task within a protocol. A clock icon with an arrow pointing into and away from it indicates that the task serves both time constraints.

Editing a time constraint

Note: The software allows you to edit time constraints during protocol runs. The time constrained tasks must be downstream from the task that is currently running. However, an edited time constraint that is too close to the running task will not likely be applied.

To edit a time constraint between two dependent tasks:

1 In the protocol, right-click one of the two dependent tasks, and then select **Edit time constraint.** The Edit Time Constraint dialog box opens. All constraints relevant to the selected task appear in the dialog box.



2 Change the Time, Plus and Minus values, and then click **OK** to save the changes and return to the protocol.

To display and edit all time constraints in a process or subprocess:

1 In the protocol, right-click the process plate or subprocess icon, and then select **Edit time constraint.** The Edit Time Constraint dialog box opens. All constraints relevant to the selected task appear in the dialog box.

2 Change the Time, Plus and Minus values, and then click **OK** to save the changes and return to the protocol.

Removing time constraints

To remove a time constraint between two dependent tasks:

In the protocol, right-click one of the two dependent tasks, and then select one of the following:

Command	Description
Edit time constraint	Allows you to remove a selected time constraint between two tasks. When you use this command, the Edit Time Constraints dialog box opens. Select the time constraint you want to remove, and then click **Remove.**
Remove all time constraints	Removes all time constraints relevant to the selected task.

Related information

For information about...	See...
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Setting up Startup and Cleanup Protocol processes

About Startup and Cleanup Protocols

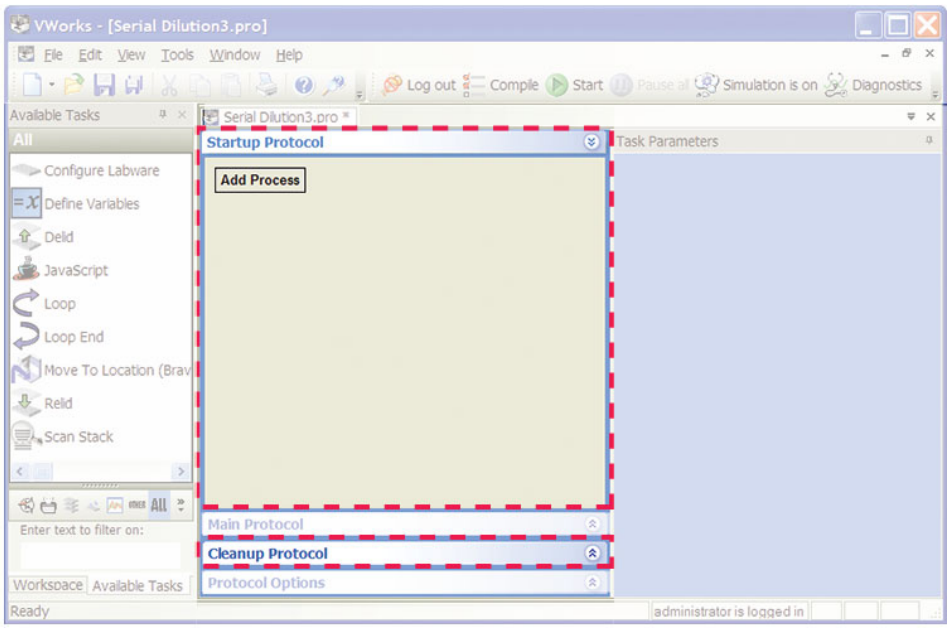
Startup Protocols are processes that are run before the Main Protocol starts. For example, you can use the Startup Protocol to prime pumps with fluid.

Cleanup Protocols are processes that are run after the main protocol finishes. For example, you can use the Cleanup Protocol to purge lines with a buffer or cleaning solution.

Procedure

To set up a startup protocol or cleanup protocol process:

1 In the protocol area, click **Startup Protocol** or **Cleanup Protocol.**



2 Add processes, set plate properties, and add tasks as you would in the Main Protocol.

Related information

For information about...	See...
Adding processes	"Adding processes" on page 58
Setting plate properties	"Setting plate parameters" on page 59
Adding tasks	"Adding and deleting tasks" on page 67

Saving the protocol

Prerequisites

You must have administrator or technician privileges to save a protocol.

VWorks Plus only. Protocols are stored as records of interest. A protocol has dependencies on other records of interest, such as a device profile, device file, labware entries, and so forth. To run a protocol successfully, all the dependent records must be in a development state equal to or greater than the protocol. For example, if the protocol is in validation, the device file must also be in a validation or released state.

Procedure

CAUTION **When you edit a protocol, the changes take effect immediately. You must save the changes before you exit the VWorks software or the changes will be lost.**

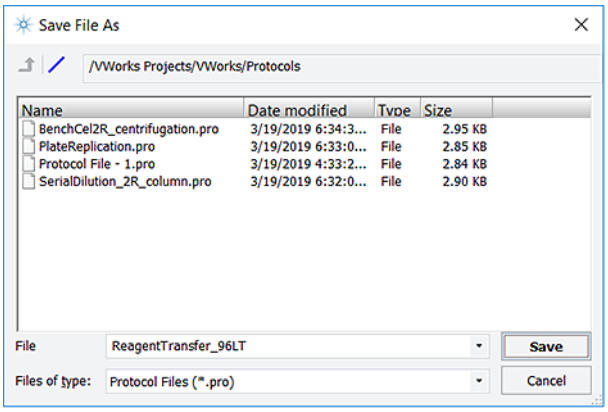
IMPORTANT You should regularly back up the protocol and associated components in case they become damaged or lost. For instructions, see "Exporting and importing protocols and associated records" on page 703.

To save a protocol:

1 Select **File > Save.**

If you just created a new protocol, proceed to step 2 to assign a name to the protocol before saving the file.

2 In the **Save File As** dialog box, type the file name, and then click **Save**.



3 *VWorks Plus only*. If an audit trail is being logged for records in the development state, the Audit Comment dialog box opens. Select or type the audit comment, and then click **OK**.

Related information

For information about...	See...
Adding notes about a protocol	"Setting protocol options" on page 42
Compiling a protocol	"Compiling the protocol" on page 77
Simulating a protocol run	"Simulating the protocol run" on page 80
Backing up protocol and associated components	"Exporting and importing protocols and associated records" on page 703

Compiling the protocol

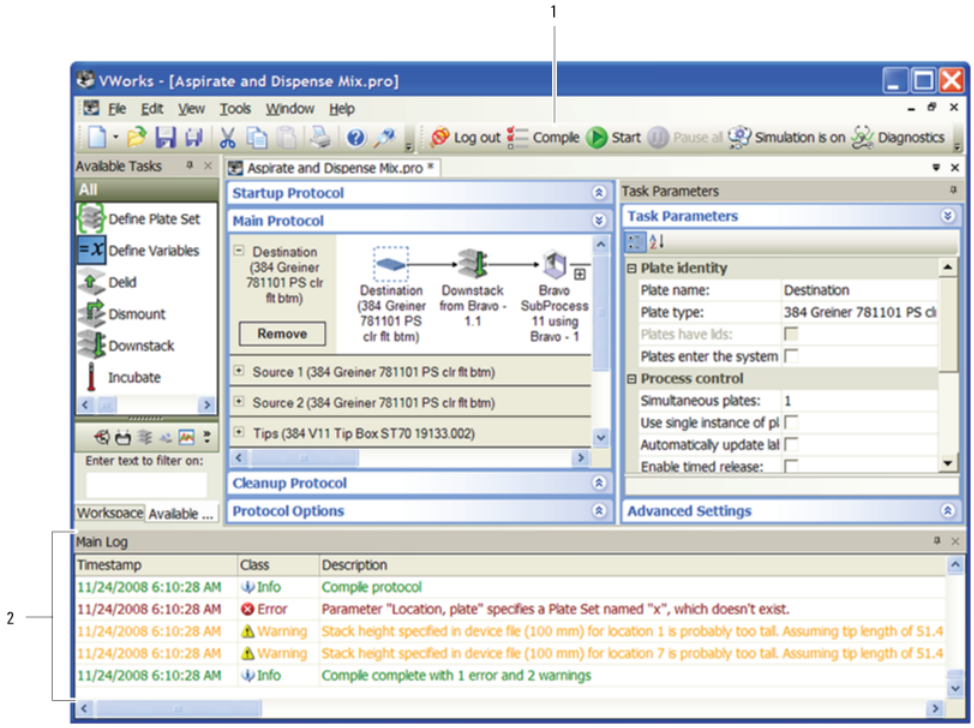
You can compile a protocol before you run it. During the compiling process, the software reports errors found in the protocol. You can use the error information to troubleshoot the protocol.

Note: The software automatically compiles the protocol whenever you start a run.

Running the compiler

To compile a protocol:

1. On the toolbar, click **Compile**. The Main Log area displays any errors found. For troubleshooting information, see "Maintenance and troubleshooting" on page 699.
2. Review the error and warning messages in the Main Log. You should fix all the errors. Depending on the protocol-writing stage, you can choose to ignore some of the warnings.

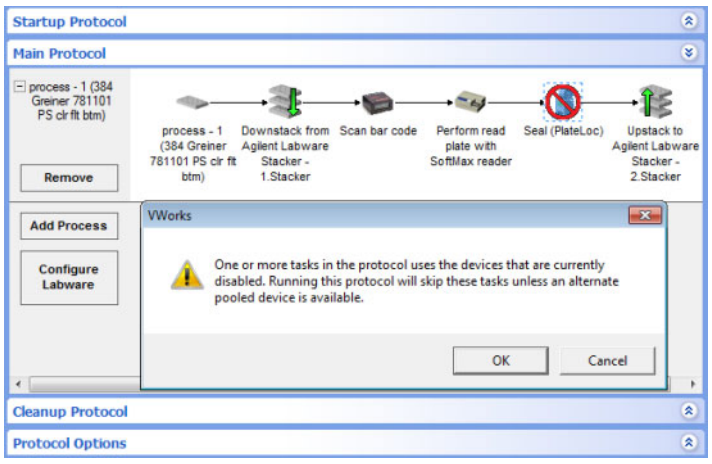


3 Repeat steps 1 and 2 until the protocol compiles error-free.

4 Save the changes you made to the protocol.

Compiling a protocol with disabled devices

When you compile a protocol that contains tasks pointing to a disabled device, and no equivalent devices are available in the pool, the software displays a warning message and a ⃠ is displayed on the affected task.



Click **OK** to compile the protocol.

For more information about temporarily disabling devices in a device file, see "Disabling and enabling a device in the device file" on page 293.

Related information

For information about...	See...
Troubleshooting compile errors	“Maintenance and troubleshooting” on page 699
Disabling a device in a device file	“Disabling and enabling a device in the device file” on page 293
Saving the protocol	“Saving the protocol” on page 76
Simulating the protocol run	“Simulating the protocol run” on page 80
Printing the protocol	“Printing protocols” on page 86

Simulating the protocol run

About run simulations

A protocol simulation is a virtual run where the software performs the tasks without moving robots or labware. Simulation runs are useful for troubleshooting scheduling and placement errors and optimizing throughput.

After checking for compiler errors, you can start a simulation to verify that tasks are completed and sequenced correctly. In addition, the simulation can help you find:

- Deadlocks
- Periods of inefficiency, such as when the robot is not being used
- Microplates spending different times at critical steps when they should be run under identical conditions
- A number of simultaneous plates that is too high or too low

Note: You can simulate the scheduling of multiple protocol runs, including those that run simultaneously.

Simulation time

The software uses the following lengths of time to simulate robot motions:

Robot movement	Time (sec)	
BenchCel robot	Slow	16
	Medium	10
	Fast	4
Phantom Robot	Slow	16
	Medium	10
	Fast	4
Task	Specified in the task Advanced Settings area. The default value is 5.0.	

Note:

- The BenchCel robot movement times are averages and might be conservative. The robot might take more or less time during a real protocol run.
- When using the Phantom Robot in a simulation, the software does not prompt the operator to move labware. Instead, the software uses the simulation time shown in the table.

You can override the default task simulation time in two ways:

- Perform a number of dry runs and allow the software to determine the average task time as it reaches a steady-state. The software then uses the average of the times during the subsequent simulation. See "Optimizing simulation time" on page 81.

- Manually set the desired task time in the Advanced Settings area of a task. See "Setting desired task times" on page 81.

Optimizing simulation time

To increase the accuracy of the run time, you can determine the optimum simulation time.

To determine the optimum task simulation time:

1. Start the simulation with the default task run times and the number of microplates expected for a run.
2. Resolve deadlocks and major errors in the protocol. Use the Gantt Chart to identify rate-limiting tasks and make adjustments to improve throughput. For instructions, see "Tracking the run progress of instances or devices" on page 330.
3. Turn off the simulation mode and perform a dry run with empty labware. Make sure the number of plate instances processed equals the number of simultaneous plates allowed. Doing so allows the run to reach a steady state and enables the software to update the task simulation times with actual times.

IMPORTANT

If the number of plate instances does not reach the number of simultaneous plates allowed, the software will not be able to update the task simulation times.

4. Save the protocol to save the updated task simulation times.

Setting desired task times

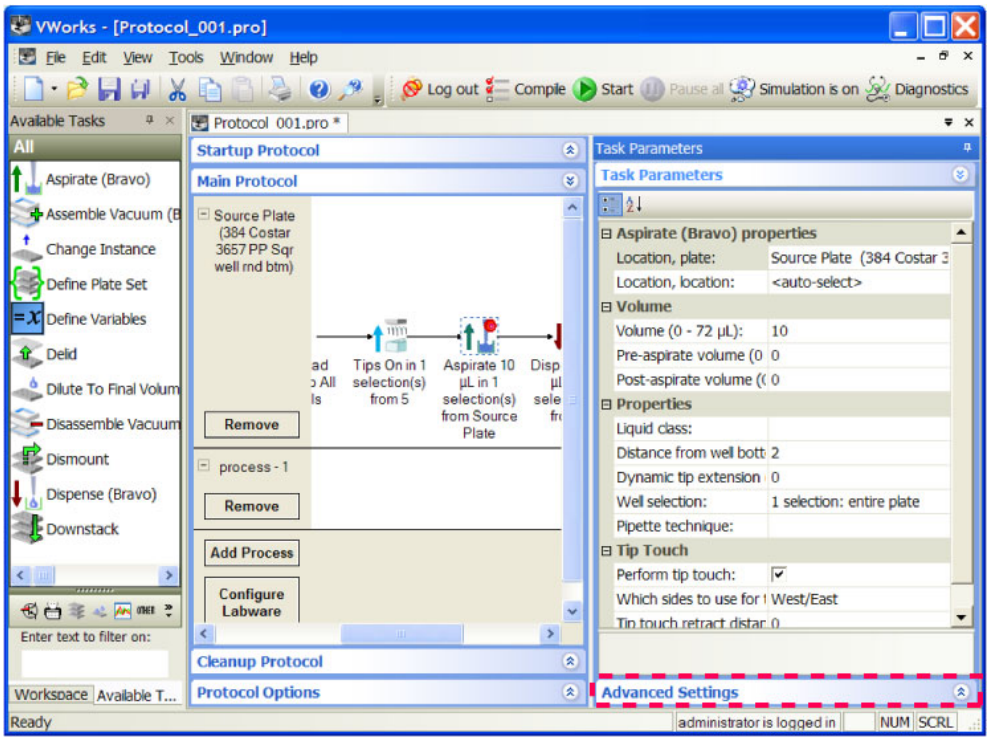
You can manually set task times to override the default task time or the optimized task time.

IMPORTANT

The manually specified time is only retained and used for simulations. The optimized time will override the manually specified time if you turn off the simulation mode and run the protocol.

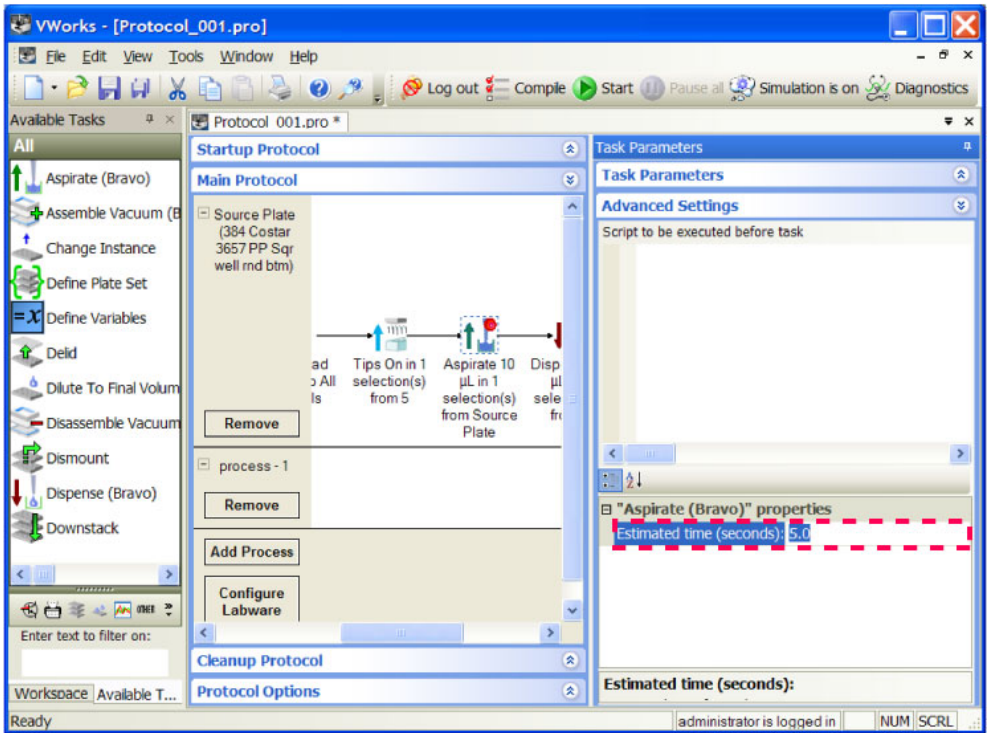
To set a desired task simulation time:

1. Select a task in the protocol.
2. In the **Task Parameters** area, click **Advanced Settings.**



3 At the bottom of the **Advanced Settings** area, view the existing value or type a new value for **Estimated time** for the selected task. To provide optimal simulation times, see "Optimizing simulation time" on page 81.

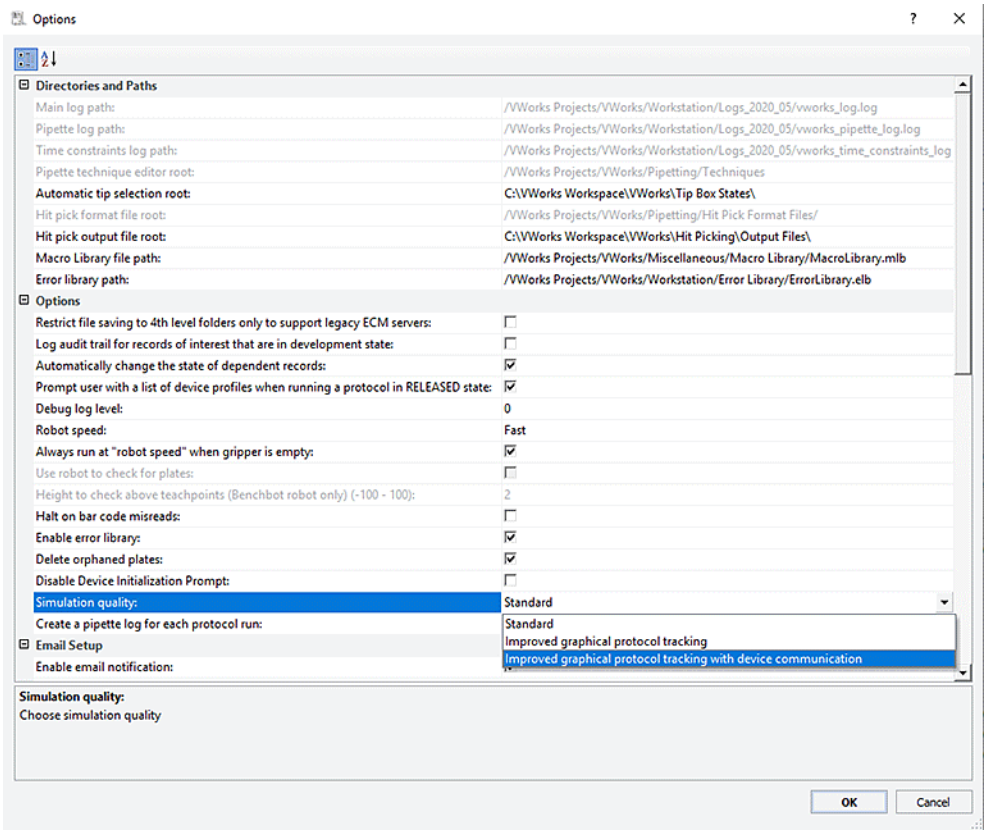
Note: Only some tasks have adjustable run times.



Running the simulator

To run the simulator:

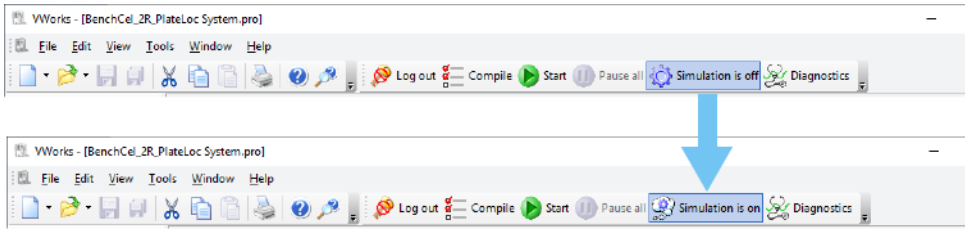
1 Select **Tools > Options.** In the **Options** dialog box, select the simulation quality you want to use:



Simulation quality	Description
Standard	Protocol tracking (green dots) is close to accurate, but not exact. This option results in faster simulation, because certain physical constraints are not simulated. In addition, less time is devoted to refreshing the graphical user interface.
Improved graphical protocol tracking	Protocol tracking is more accurate but results in longer simulation time than the Standard quality option. Similar to the Standard quality option, the software does not query the device plugins for physical constraints. However, the software ensures the green dots in the Protocol area are positioned accurately during the simulation.

Simulation quality	Description
Improved graphical protocol tracking with device communication	Protocol tracking is the most accurate. The software queries the device plugins to ensure every move is physically permissible, resulting in a slower but more accurate simulation.

2 Turn on the simulation mode: Click **Simulation is off** on the toolbar. The button changes to **Simulation is on.**

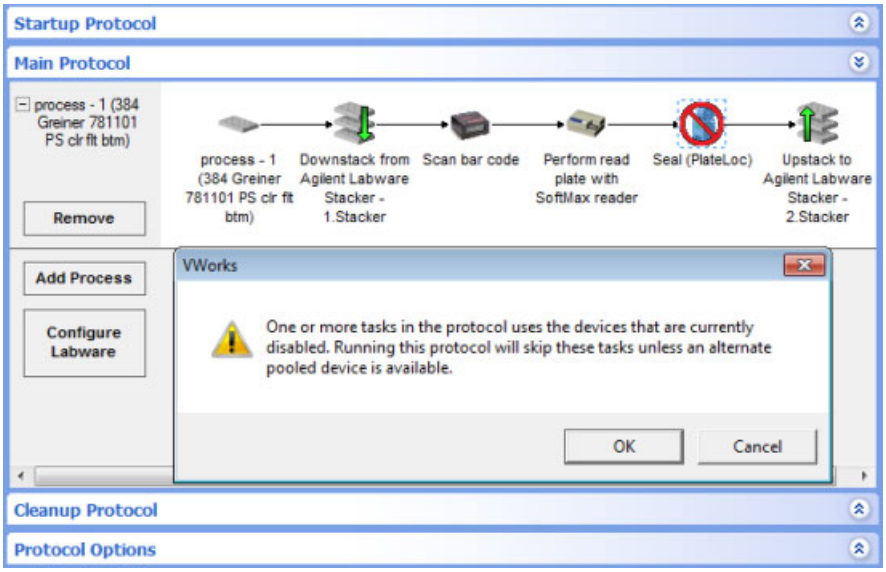


3 Start the protocol run. See "Starting the protocol run" on page 312 for the different ways you can start a run.

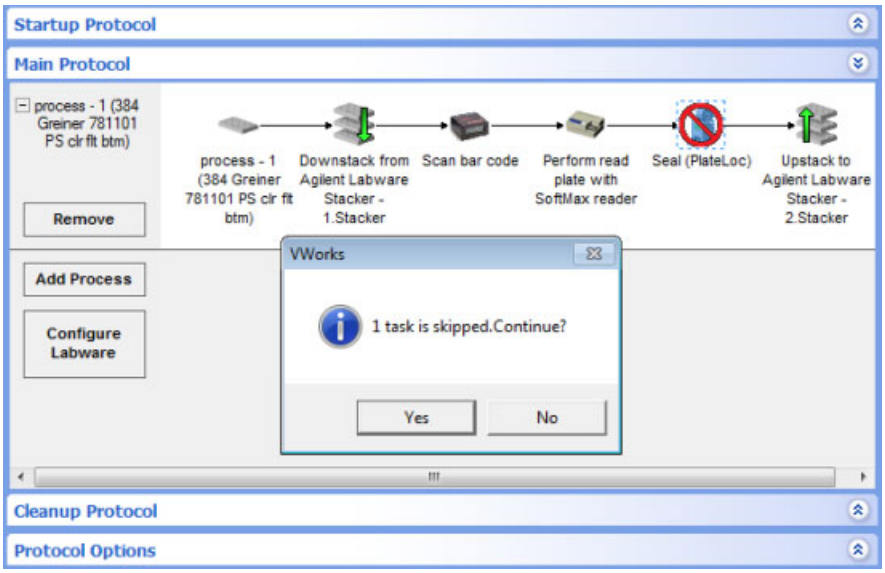
The simulation starts. The Run Configuration Wizard and other dialog boxes open as they would in a real protocol run. Follow the instructions in the dialog boxes to proceed with the simulation. For detailed information about Run Configuration Wizard, see "Starting the protocol run" on page 312.

Simulating a run with a disabled device

When you run a protocol in simulation that contains tasks pointing to a disabled device, and no equivalent devices are available in the pool, the software displays a warning message and a ⃠ is displayed on the affected task.



Click **OK** to run the protocol. The following dialog box appears indicating how many tasks will be skipped:



Click **Yes.** The Run Configuration Wizard opens. Follow the instructions in the wizard. For details on the wizard, see "Starting the protocol run" on page 312. During the run, the tasks associated with the disabled device are skipped.

Related information

For information about...	See...
Compiling the protocol	"Compiling the protocol" on page 77
Selecting simulation qualities	"Setting general and view options" on page 301
Changing the robot speed	"Setting general and view options" on page 301
Temporarily disabling a device in a device file	"Disabling and enabling a device in the device file" on page 293
Scheduling multiple runs	"Starting the protocol run" on page 312
Managing runsets	"Managing runsets" on page 317
Monitoring runs	"Monitoring the overall run progress" on page 329
Pausing runs	"Pausing the run" on page 337
Deadlock recovery	"Recovering from deadlocks" on page 730

Printing protocols

You can use a printout of your protocol to:

- Troubleshoot the protocol
- Present in a report

Before you start

Make sure:

- The computer is connected to the printer you want to use.
- The printer driver is installed on the computer that has VWorks software installed.

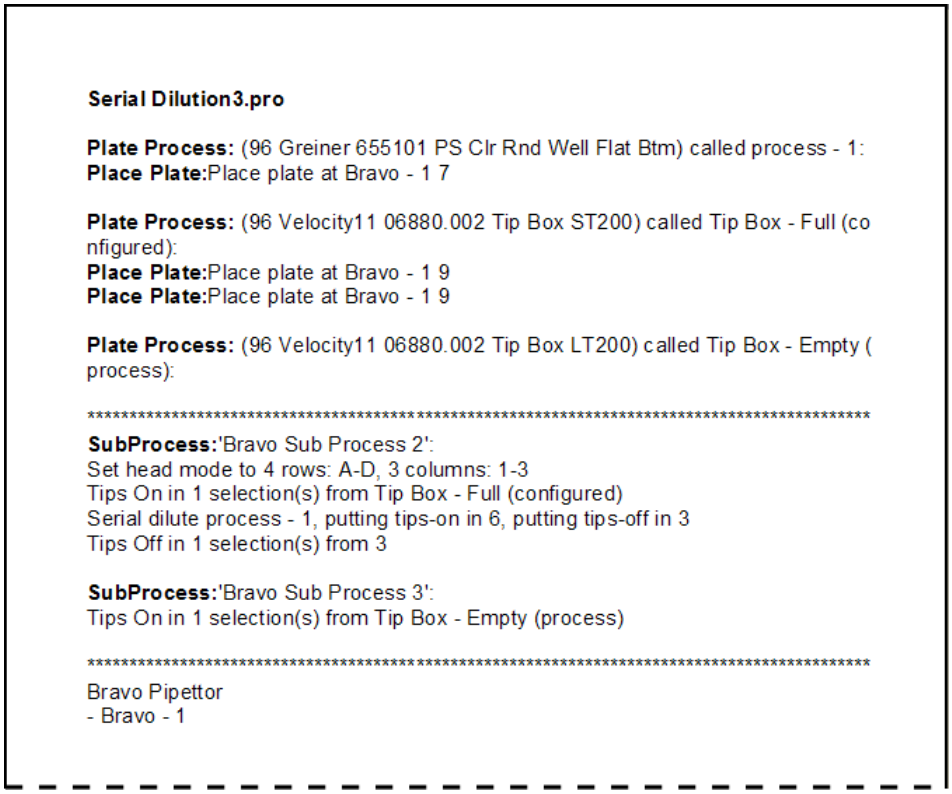
Procedure

To print a protocol:

1 Open the protocol you want to print.

2 In the VWorks window, select **File > Print.**

The protocol printout shows the following:



Section	Description
Protocol name	The name of the protocol.

Section	Description
Process name	The name of the process and the associated labware type.
Tasks	The tasks in the process, the devices used in the task, and location information.
Subprocesses	The subprocess and all tasks in the subprocess.

Related information

For information about...	See...
Compiling protocols	“Compiling the protocol” on page 77
Simulating protocol runs	“Simulating the protocol run” on page 80
Troubleshooting problems	“Maintenance and troubleshooting” on page 699

3 Creating a protocol: advanced topics

Read this chapter if you are an administrator or technician who writes protocols.

This chapter contains the following topics:

- "Tracking barcodes" on page 90
- "Using simple variables" on page 93
- "Using JavaScript" on page 100
- "VWorks-defined functions" on page 103
- "Experiments database objects" on page 111
- "InventoryLabware object" on page 118
- "plate object" on page 120
- "plates[ ] object" on page 123
- "task object" on page 126
- "Using JavaScript with the CentrifugeAuto task" on page 136
- "About scripting the Print and Apply task" on page 137
- "plateDB object" on page 140
- "runset object" on page 148
- "forms object" on page 151
- "Using start and finish protocol scripts" on page 152
- "Using JavaScript utilities" on page 153
- "Creating new protocol tasks using the JavaScript Wrapper" on page 158

Tracking barcodes

About this topic

The VWorks software allows you to track barcodes that are read or applied at a device. Barcode readers can be installed on a number of devices in the lab automation system. For example, a barcode reader can be installed on a platepad so that every time a labware is placed there, the barcode is automatically read.

If your lab automation system contains a Microplate Labeler, you have the ability to print and apply barcode labels on microplates. If the Microplate Labeler includes a barcode reader, barcode labels can be read.

To track barcodes in the system, you need the following:

- Input files
- Data files (See "Data files" on page 91.)

Input files

Description

Created by upstream applications such as LIMS or created manually, barcode input files can be used to verify barcodes on incoming labware. As barcoded labware enter the system through tasks such as Downstack, the barcodes are scanned and compared against the barcodes in the barcode input file. You specify the input file to use when you set the process plate parameters. You can also specify a device to use to quarantine plates for which the barcode label is misread or incorrect. For details see "Setting plate parameters" on page 59.

Note: The software remembers where you are in the input file. For example, if you have 10 entries in the file, and the first protocol run processes the first six microplates, the next run will start at entry seven. To reset the file to the first row, reload the file. Barcode files are reloaded when you open a protocol file, when you update the Barcode file directory in Protocol Options, or when you update the Barcode filename or Has header option when setting up the plate parameters.

Requirements

Input files must meet the following requirements:

- The file must be a comma-separated value format with the .bar, .csv, or .txt file name extension.
- *Optional*. The file can contain a header that describes the columns in the file (for example, Plate Barcode).
- The file must contain at least one column.
- One of the columns must list the barcodes.

The following example shows an input file that is displayed in Excel. The file contains a header that labels two columns: Plate ID and Barcode.

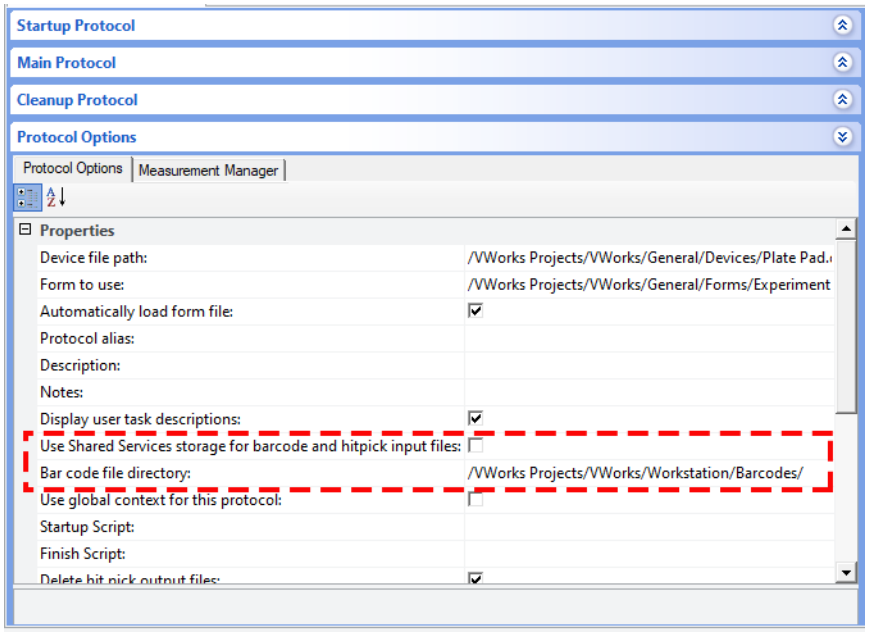
	A	B
1	Plate ID	Barcode
2	Plate001	100001
3	Plate002	100002
4	Plate003	100003
5	Plate004	100004
6	Plate005	100005
7	Plate006	100006
8	Plate007	100007
9	Plate008	100008
10	Plate009	100009
11	Plate010	100010
12	Plate011	100011

Storage location

You specify the following barcode file storage location options in the Protocol Options area of the VWorks window:

- Use Shared Services storage for barcode and hitpick input files
- Barcode file directory

The following figure shows an example of the barcode file settings for storage location. For details on these settings, see "Setting protocol options" on page 42.



Data files

Description

Created by upstream applications such as LIMS or created manually, barcode data files are used by the Print task or the Print and Apply task to:

- Print barcodes on labels. For example, the software reads a row in the file and prints the barcode presented in that row.

- Look up barcodes to print on labels. For example, the software reads the east-side barcode on a labware, looks up the barcode in the file, and prints the north-side barcode that is in the corresponding column (same row).

You specify the data file to use when you specify the task parameters for the Print task or Print and Apply task. See "Print" on page 658 and "Print and Apply" on page 381.

Requirements

Barcode data files must meet the following requirements:

- Be a comma-delimited text file with the .csv file name extension
- *Optional.* Contain a header, which can be in any format.
- Contain at least one column. For example, the file can contain four columns, each representing a side of the microplate.

The following example shows a data file displayed in Excel. The file contains four columns, each representing a side of the microplate. In addition, the file contains a header in row 1.

	A	B	C	D
1	North	South	East	West
2	A00001	B00001	C00001	D00001
3	A00002	B00002	C00002	D00002
4	A00003	B00003	C00003	D00003
5				

Use the **File > Import Misc File** command to import the data file into the Shared Services storage.

Related information

For information about...	See...
Importing barcode files	"Importing and exporting miscellaneous files" on page 49
Microplate Labeler	*Microplate Labeler User Guide*
Print task	"Print" on page 658
Print and Apply task	"Print and Apply" on page 381

Using simple variables

About this topic

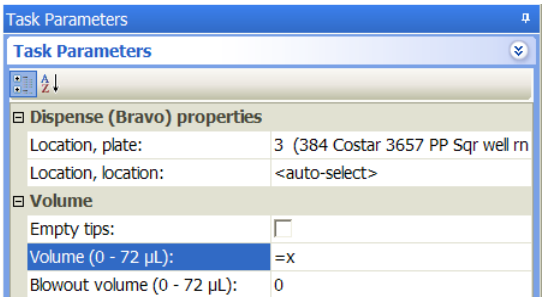
This topic presents the following:

- "About variables" on page 93
- "Variable and snippet syntax" on page 94
- "Defining variables and adding code snippets" on page 94
- "Adding user message prompts" on page 98

About variables

You can use variables to assign operator-supplied values to task parameters, provide an initial value for looping, or reference an array in another task. An example of variable usage is as follows: During a protocol run, the software will prompt the operator for aspirate and dispense volumes. The software will use the operator-supplied values during the run.

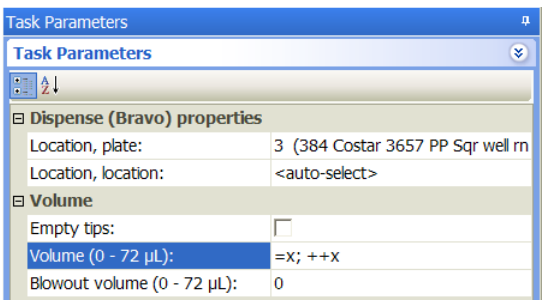
Instead of writing detailed JavaScript code in the JavaScript task Advanced Settings area, you can type a simple variable assignment and a code snippet directly in a field in the Task Parameters area. The following example shows how to assign a variable called x to the Dispense Volume parameter in the Task Parameters area.



Assigning the variable in the Task Parameters area is equivalent to providing the following line of code in the Advanced Settings area:

```
task.Volume = x;
```

You can also add a snippet of code after the variable assignment. For example, you can assign the Dispense Volume to x, and then increment it by the same amount in each loop during the protocol run.



To prompt the operator for the values to use during the protocol run, you can select the option in the Define Variables task or add User Message tasks at the desired points in the protocol. See "Adding user message prompts" on page 98.

Variable and snippet syntax

When adding a variable and code snippet in the Task Parameters area, use the following syntax:

```
=x; <code snippet>
```

where `x` is the variable name and `<code snippet>` is additional code you want to run during the task. Use standard JavaScript rules for the variable name and for the code snippet.

Note: Variables defined here must be predefined in the protocol in the Define Variables task, User Message task, a previous JavaScript code snippet, or in the Advanced Settings area.

Defining variables and adding code snippets

You can define variables using the following:

- *Define Variables task*. In the Define Variables task, you can define multiple variables and set initial values. For more information, see "Define Variables" on page 645.
- *Define Plate Set task*. In the Define Plate Set task, you can define a variable that identifies a group of process plates to be processed identically. For more information, see "Define Plate Set" on page 640.
- *Task parameter variables*. You can define a variable for almost any parameter. For example, in the Aspirate task, you can define a variable for the Volume parameter. For more information, see "Task parameter variables" on page 94.
- *Startup Protocol JavaScript task*. Using the JavaScript task in the Startup Protocol, you can define multiple variables and assign initial values to them. For more information, see "Startup Protocol JavaScript task" on page 96.

Task parameter variables

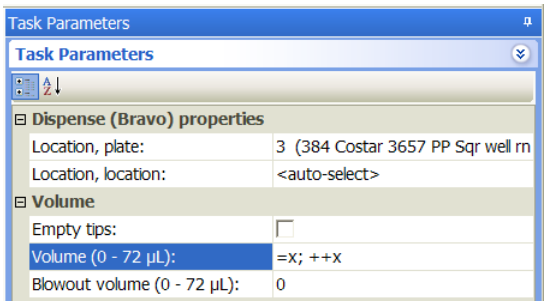
The way you define a variable for other tasks depends on the field input style:

- Text box format
- List format
- Time format

Note: Parameters that require fixed values do not accept variables. For example, passwords, deck locations, and IP addresses cannot be assigned the value of a variable.

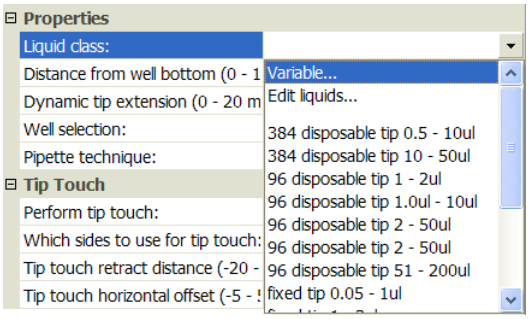
To assign a variable to a parameter whose value is displayed in a text box:

In the parameter value text box, type the variable assignment and optional code snippet. For syntax requirements, see "Variable and snippet syntax" on page 94.

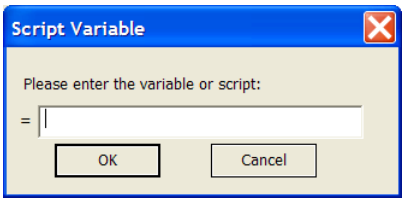


To assign a variable to a parameter whose value can only be selected from a list:

1 In the list, select **Variable.**



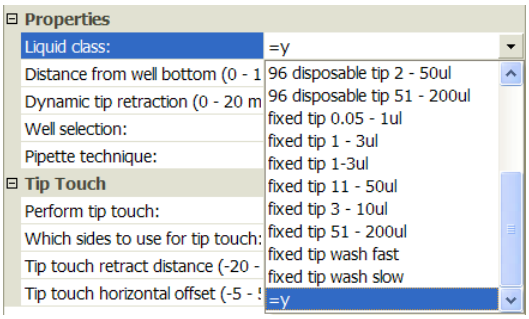
The Script variable dialog box opens.



2 Type the variable assignment and optional code snippet. For syntax requirements, see "Variable and snippet syntax" on page 94.

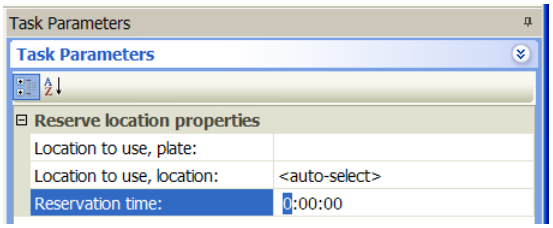
Note: The = symbol in front of the text box indicates that the software will automatically add the = symbol in front of the variable.

3 Click **OK.** The new variable and optional code snippet appear in the parameter value list.

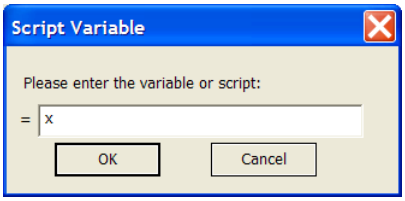


To add a variable to a parameter whose value is in the time format (h:mm:ss):

1 In the parameter value box, type = in the h, mm, or ss field. The Script variable dialog box opens.

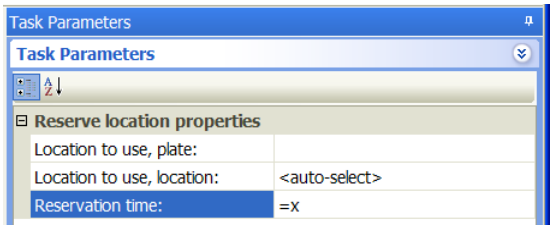


2 Type the variable assignment and optional code snippet. For syntax requirements, see "Variable and snippet syntax" on page 94.



Note: The = symbol in front of the text box indicates that the software will automatically add the = symbol in front of the variable.

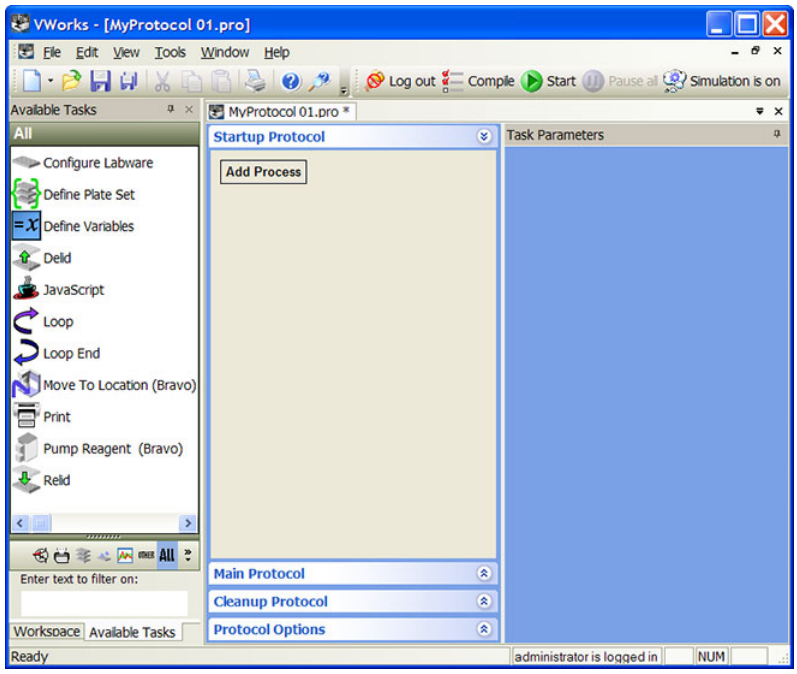
3 Click **OK.** The new variable and optional code snippet appear in the parameter field.



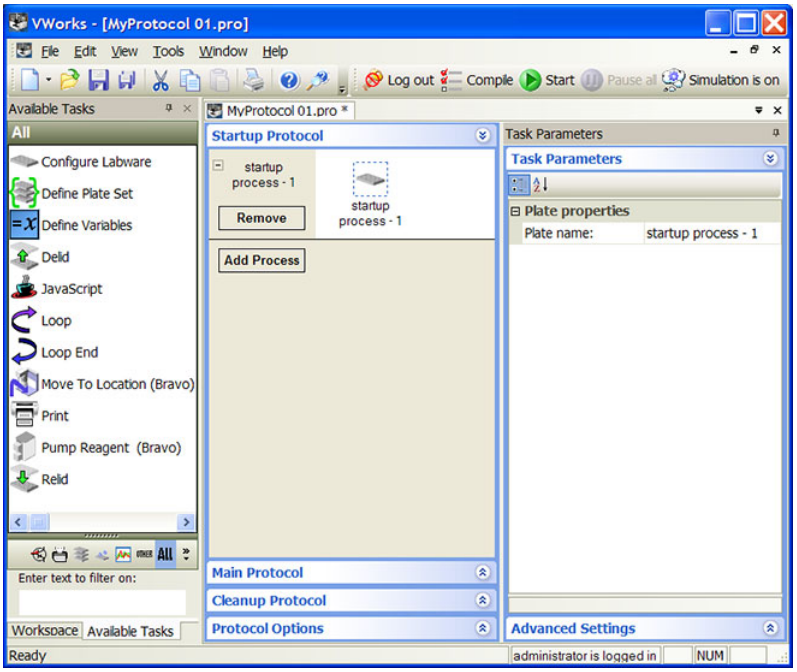
Startup Protocol JavaScript task

To create multiple variables and assign initial values to them:

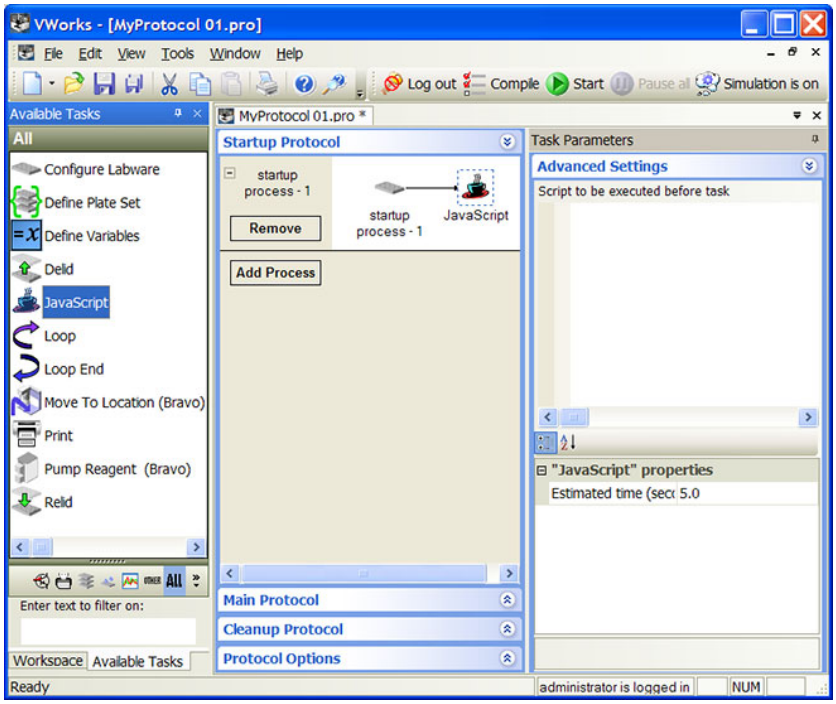
1 In the protocol, click **Startup Protocol.**



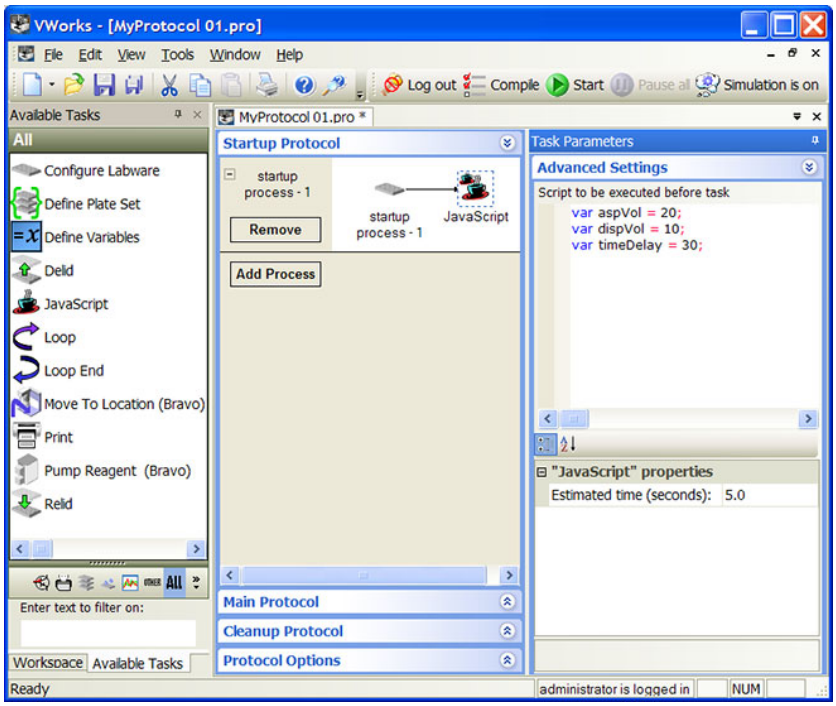
2 Click **Add Process.** A Startup process appears.



3 Add the **JavaScript** task.



4 In the **Advanced Settings** area, type the variable assignments.



Adding user message prompts

After adding variables to the desired task parameters, you can add User Message tasks to prompt operators for values at the beginning of the protocol run or at the desired points during the protocol run.

Note: You do not need to add the User Message task when using the Define Variables task. The Define Variables task contains an option to add a user message prompt.

To add User Message tasks:

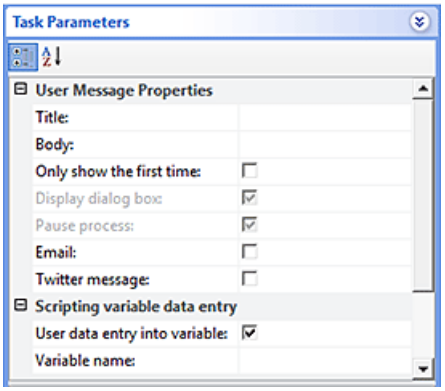
1 In the Startup Protocol:

- Add a User Message task for each variable assigned in a task parameter input field.
- Add a User Message task for any variable that you assigned in the Advanced Settings area of a given task. If you added variables in the Advanced Settings area, add the User Message task after the JavaScript task in the startup protocol.

Alternatively, at the desired points in the Main Protocol, add a User Message task for each variable assigned in a task parameter input field.

2 In the **Task Parameters** area, type values and select the desired options in the **User Message Properties** table. For a description of the properties, see "User Message" on page 682.

3 Select **User data entry into variable** and type the name of the variable in the **Variable name** box. During the protocol run, the software will prompt the operator for a value and assign it to this variable.



IMPORTANT

If you added multiple variables in the startup protocol, do not use the Variable name property. Instead, use the Body property to instruct the operator to set the variable values in the Advanced Settings area. For details on the Body property, see "User Message" on page 682.

Related information

For information about...	See...
User Message task	"User Message" on page 682
JavaScript task	"JavaScript" on page 653
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
Using JavaScript utilities	"Using JavaScript utilities" on page 153

For information about...	See...
Startup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

Using JavaScript

About this topic

This topic explains how to use JavaScript in the VWorks software. For a full description of the JavaScript language, see the Mozilla Developer Center at https://developer.mozilla.org/en-US/docs/Web/JavaScript.

Read this topic if you have administrator or technician privileges. This topic assumes that you know how to write programs in JavaScript or have basic programming knowledge.

How JavaScript is used in the software

A comprehensive JavaScript engine is implemented in the VWorks software so that you can customize the software as follows:

- Change the existing task parameters.
- Skip a task if certain conditions are met.
- Repeat a task if certain conditions are met.

The task parameters can change dynamically during a run based on conditions such as:

- Information passed from an external source, such as a database
- The number of times the protocol has cycled
- Feedback on changing conditions or data values during the run

Examples of use

You can use JavaScript in the VWorks software to:

- Print the parameters of a task to the main log.
- Run a command that launches an external application, such as a batch file or database-updating program.
- Performing file input and output operations.

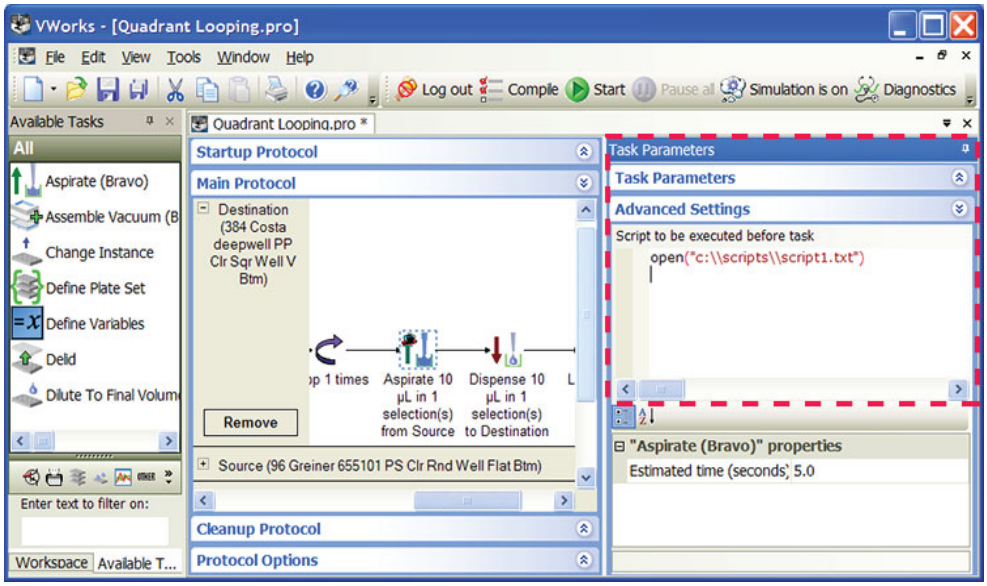
Where to write JavaScript

You can write JavaScript in the Advanced Settings area of any task or in the JavaScript task in the Startup, Main, and Cleanup Protocols. JavaScript written in any task is run differently than code written in the JavaScript task:

- *Any task*. During a protocol run, the software will run the JavaScript first before evaluating whether the task is possible.

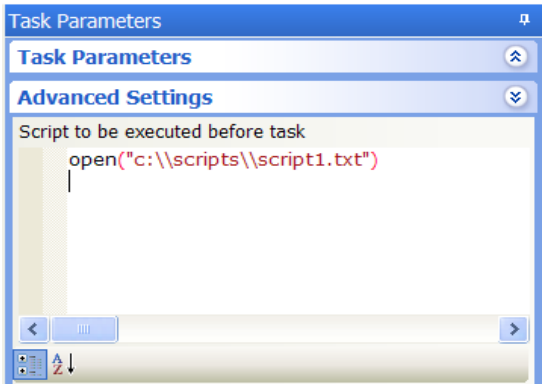
- *JavaScript task*. You can add the JavaScript task to run a program that is independent of any task. During a protocol run, the software will run the JavaScript as it reaches the JavaScript task.

The following example shows JavaScript written in the Advanced Settings area of the Aspirate task. Notice that the script is written directly in the text box.



You can add JavaScript in the following ways:

- Directly in the text box in the Advanced Settings area (see the previous example)
- As an external file that is called by the open( ) function in the text box



About JavaScript variables

By default, the values of all variables are cleared (set to undefined) before the next protocol is run. To retain the value of all variables from protocol to protocol, select the **Use global context for this protocol** option in the Protocol Options tab. When the option is turned on, a variable and its value assigned in one protocol can be used by other protocols that have the same variable until you exit the software. If using the global context option, the values of variables are not reset until the software is restarted.

CAUTION **If you select the Use global context for this protocol option, variables with the same name in different protocols will overwrite each other.**

Cautions

Before running a protocol containing JavaScript, compile the protocol and run it in simulation mode.

During the compiling process, the software uses values displayed in the Task Parameters area and not the values set by JavaScript. Therefore, some errors might not be detected during compilation. The values that appear in the Task Parameters area do not reflect the effects of the JavaScript.

Be aware that JavaScript does not check pipetting volumes before the run begins. Therefore, you must make sure that the pipetting steps make logical sense. For example, the software does not alert you if a JavaScript attempts to aspirate 1 µL from a microplate well that can only hold 0.5 µL.

Related information

For information about...	See...
JavaScript language	Mozilla Developer Center
Using script variables directly in task parameters	"Using simple variables" on page 93
JavaScript task	"JavaScript" on page 653
Using JavaScript utilities	"Using JavaScript utilities" on page 153
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

VWorks-defined functions

The VWorks JavaScript interpreter supports the JavaScript 1.5 or later core functions and objects. VWorks also has its own functions and objects that can be used.

The following VWorks-defined functions are available globally, so they are not restricted to a particular object or programming context.

Function	Description
DisplayAsyncMessageBox().	Designed to be called from a script that is run from a button on a form. For details, see "Opening a message from a form button using JavaScript" on page 247.

DownloadFromStorage()	Downloads a file from Shared Services (OLSS) storage to a specified Windows file path. Writes a JavaScript error message to the Main log if the download fails.
	Files that are created external to the VWorks software, for example, .xlsm files, cannot be run directly from OLSS storage. However, you can create a script using DownloadFromStorage() to download the file from OLSS storage to the local file system, ShellExecute() to run the downloaded file, and UploadToStorage() to upload the file again to OLSS storage.
	Parameters:
	1 *Text string*. Shared Services file path to the file name to be downloaded, where the file path
	• May or may not begin with "[olssvr]:".
	• Must be relative to the Shared Services root directory for the software edition:
	`"[olssvr]:VWorks Projects/VWorks/`
	Note: An extra / or \ at the beginning of the path or after [olssvr]: is ignored.
	2 *Text string*. Windows file path to the directory where the file is to be downloaded.
	The file will be downloaded with the same file name that it has in Shared Services storage.
	Examples:
	`DownloadFromStorage("[olssvr]:VWorks Projects/VWorks/Workstation/Barcodes/MyBarcodes.csv","C:/temp/Myfiles/")`
	`DownloadFromStorage("[olssvr]:VWorks Projects/VWorks/Miscellaneous/Scripts/Serial Dilution Method Setup Tool.xlsm","C:\\Temp");`
	`ShellExecute("C:\\Temp\\Serial Dilution Method Setup Tool.xlsm","",true);`
	`UploadToStorage ("C:\\Temp\\Serial_Dilution_Method_File.csv","[olssvr]:VWorks Projects/VWorks/Workstation/Hitpick Input Files/");`

Function	Description

GetGlobalObject()	Provides the ability to get the global context from a JavaScript that is executed in a context other than the global context. Returns an object that has all the variables of the global context. For example, in a protocol that uses the local context for variables, you can have a JavaScript task that includes a variable in the global context.
	Example:
	`//(0) Create a reference to global object from this context.`
	`var g_ref = GetGlobalObject();`
	`//(1) Declare a var xx in GlobalObject.`
	`g_ref.xx = 4`
	`//(2) Increment the variable.`
	`g_ref.xx++`
	`//(3) Print the variable.`
	`print(g_ref.xx)`
	Note: In this example, lines (2) and (3) could be executing from different protocols, each in their own private context, with shared data results.
	Note: You can set a breakpoint on the task that uses this function to open the Debugger dialog box and view the variable (g_ref). For details on setting breakpoints, see "Using breakpoints to monitor and troubleshoot tasks" on page 724.

IsCompliantMode()	Returns a Boolean:
	• If true, VWorks is running in Compliance mode (compliance-enabled edition).
	• If false, VWorks is running in Non-Compliance mode (Standard edition).
	Example:
	`var isCompliant = IsCompliantMode();`
	`if (isCompliant)`
	`  print ("Running in Compliant mode");`
	`else`
	`  print ("Not running in Compliant mode");`

Function	Description

open()	Opens a text file and immediately executes the file contents as script.
	Parameter: Text string
	Make sure the file is in ANSI text format. The file extension is not important.
	Note: Microsoft Notepad saves in ANSI text format by default.
	Examples (local file system):
	`open("c:\\VWorks workspace\\script.js")`
	`open("c:\\VWorks workspace\\script.txt")`
	Example (Shared Services storage):
	`open("[olssvr]:VWorks Projects/VWorks/Miscellaneous/Scripts/script.js")`
	Note: You can import JavaScript (.js) and text (.txt) files into Shared Services storage using the VWorks File > Import Misc command.

print()	Prints time-stamped messages to the VWorks log. Parameter: Text string Example: `print(plate.name)`

run()	Runs a Microsoft Windows script as though it is being called from a command line.
	Note: The run() function cannot invoke Microsoft Excel if passed the name of an .xlsx file.
	Writes a JavaScript error message to the log if the operation fails.
	Parameters:
	• *Text string*. Required. Allows you to initiate a command that you could otherwise enter into the Windows Run dialog box, such as notepad text.txt (opens a file named text.txt in Windows Notepad).
	• *Boolean true/false*. Optional.
	– *False*. (Default) The software does not wait for the function to complete before continuing.
	– *True*. The software waits for the function to complete before continuing (blocking).
	Examples:
	`run("notepad")`
	`run("notepad",true)`
	The first example starts Notepad. The second example starts Notepad and pauses the VWorks software until you exit Notepad.
	Note: A batch file (.bat) can be run only if it resides on the local disk.

Function	Description

GetGlobalObject()	Provides the ability to get the global context from a JavaScript that is executed in a context other than the global context. Returns an object that has all the variables of the global context. For example, in a protocol that uses the local context for variables, you can have a JavaScript task that includes a variable in the global context.
	Example:
	`//(0) Create a reference to global object from this context.`
	`var g_ref = GetGlobalObject();`
	`//(1) Declare a var xx in GlobalObject.`
	`g_ref.xx = 4`
	`//(2) Increment the variable.`
	`g_ref.xx++`
	`//(3) Print the variable.`
	`print(g_ref.xx)`
	Note: In this example, lines (2) and (3) could be executing from different protocols, each in their own private context, with shared data results.
	Note: You can set a breakpoint on the task that uses this function to open the Debugger dialog box and view the variable (g_ref). For details on setting breakpoints, see "Using breakpoints to monitor and troubleshoot tasks" on page 724.

IsCompliantMode()	Returns a Boolean:
	• If true, VWorks is running in Compliance mode (compliance-enabled edition).
	• If false, VWorks is running in Non-Compliance mode (Standard edition).
	Example:
	`var isCompliant = IsCompliantMode();`
	`if (isCompliant)`
	`  print ("Running in Compliant mode");`
	`else`
	`  print ("Not running in Compliant mode");`

Function	Description

open()	Opens a text file and immediately executes the file contents as script.
	Parameter: Text string
	Make sure the file is in ANSI text format. The file extension is not important.
	Note: Microsoft Notepad saves in ANSI text format by default.
	Examples (local file system):
	`open("c:\\VWorks workspace\\script.js")`
	`open("c:\\VWorks workspace\\script.txt")`
	Example (Shared Services storage):
	`open("[olssvr]:VWorks Projects/VWorks/Miscellaneous/Scripts/script.js")`
	Note: You can import JavaScript (.js) and text (.txt) files into Shared Services storage using the VWorks File > Import Misc command.

print()	Prints time-stamped messages to the VWorks log. Parameter: Text string Example: `print(plate.name)`

run()	Runs a Microsoft Windows script as though it is being called from a command line.
	Note: The run() function cannot invoke Microsoft Excel if passed the name of an .xlsx file.
	Writes a JavaScript error message to the log if the operation fails.
	Parameters:
	• *Text string*. Required. Allows you to initiate a command that you could otherwise enter into the Windows Run dialog box, such as notepad text.txt (opens a file named text.txt in Windows Notepad).
	• *Boolean true/false*. Optional.
	– *False*. (Default) The software does not wait for the function to complete before continuing.
	– *True*. The software waits for the function to complete before continuing (blocking).
	Examples:
	`run("notepad")`
	`run("notepad",true)`
	The first example starts Notepad. The second example starts Notepad and pauses the VWorks software until you exit Notepad.
	Note: A batch file (.bat) can be run only if it resides on the local disk.

Function	Description

ShellExecute()	Runs an executable (.exe) or opens a document file as though being called from a Windows command line.
	
	Writes a JavaScript error message to the log if the attempt to execute fails.
	
	Use ShellExecute() instead of run() to *execute* a document by having Windows launch the application associated with that document type. For example, you could use this function to launch Microsoft Excel (if installed) and load an .xlsx file.
	
	Note: To launch an app with explicit command-line arguments, they need to be passed in separately as the (optional) second argument to the function.
	
	Parameters:
	• *Text string*. Required. The name of the file to be executed (.exe or document). If the string specifies
	– An .exe file, it executes as if the file name had been passed to run().
	– A file with an associated application, for example an .xlsx file on a computer where Microsoft Excel is installed, the function launches the application with the file name passed in as the document to open.
	• *Text string*. Optional. Any command-line arguments to be passed in.
	• *Boolean true/false*. Optional.
	– *True*. The software waits for the spawned process to finish before continuing (blocking).
	– *False*. The software does not wait for the spawned process to finish before continuing.
	• *Integer*. Optional. If this parameter is present, the app will be launched as follows, depending on the value of the integer. Use one of the following values:
	– 3, the app will start out maximized, if it is capable of being maximized.
	– 1 or 5, the app will start out in a normal, not-minimized state, if it is capable of displaying a top-level window.
	– 6, the app will start out minimized, as though this parameter were absent.
	Other values give undefined behavior, but do not generate errors or warnings.
	
	Examples:
	`ShellExecute ("C:\\VWorks Workspace\\VWorks\\Hit Picking\\EV2_hit_pick.xlsx")`
	`ShellExecute("C:\\Windows\\Notepad.exe","MyFile.txt",true,5)`
	
	Note: ShellExecute() can only launch apps and documents that exist in the regular Windows file system. To launch something stored in Shared Services storage (OLSS), you first call DownloadFromStorage() and supply a directory where you want to execute it from, then call ShellExecute() (or run()) on the downloaded file.

Function	Description

UploadToStorage()	Uploads a record of interest (.roiZip), which is exempt from record states, to Shared Services (OLSS) storage.
	Writes a JavaScript error message to the log if the upload fails.
	Parameters:
	1 *Text string*. Windows file path to the file to be uploaded.
	2 *Text string*. The Shared Services path of the file name to be uploaded, where the file path
	• May or may not begin with "[olssvr]:".
	• Must be relative to the Shared Services root directory for the software edition:
	`"[olssvr]:VWorks Projects/VWorks/"`
	An extra / or \ at the beginning of the path, or after [olssvr]:, is ignored.
	Note: This parameter is different from DownloadFromStorage(), where the second parameter is a destination directory. In UploadToStorage(), the second parameter is the destination path with file name. This allows UploadToStorage() to save the file to a different file name than the one it is being copied from.
	3 *Text string*. The audit trail comment (reason for the change) for the created record of interest.
	4 *Boolean true/false*:
	– *True*. The software "overwrites" a file with the same file name if it already exists. The overwritten file is assigned a new version and is retained in OLSS storage along with the newly uploaded file.
	– *False*. If a file with the same file name already exists, the software will reject the upload.
	Note: Parameter 3 may be blank for VWorks Standard edition.
	Examples:
	`UploadToStorage("C:\\VWorks Workspace\\Myfiles\\MyBarcodes.csv","[olssvr]:VWorks Projects/VWorks/Workstation/Barcodes/MyBarcodes.csv","my audit comment",false)`
	`UploadToStorage("C:\\VWorks Workspace\\Myfiles\\MyBarcodes.csv","[olssvr]:VWorks Projects/VWorks/Workstation/Barcodes/MyBarcodes.csv","",true)`

Related information

For information about...	See...
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined plate objects, properties, and methods	"plate object" on page 120

For information about...	See...
VWorks-defined task objects, properties, and methods	"task object" on page 126
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75
Using script variables directly in task parameters	"Using simple variables" on page 93
How to specify the JavaScript context for a protocol	"Setting protocol options" on page 42
JavaScript context and form design	"Understanding JavaScript context in form design" on page 243

Experiments database objects

IMPORTANT

You must have an established connection to the Experiments database to use the experimentTagsDB object. For setup and connection instructions, see the *VWorks Automation Control Setup Guide*.

About experimentTagsDB

The VWorks JavaScript interpreter provides an experimentTagsDB object that can be accessed by a script and used to query records from, add records to, and edit records in the Experiments database.

To handle most interactions with the Experiments database, you can use the built-in features in a VWorks form and the Step Started/Completed task in the protocol instead of the experimentTagsDB object.

experimentTagsDB has no properties.

About protocolExperimentData

Protocols run by a VWorks form

The JavaScript object protocolExperimentData is created automatically when a VWorks form that has an Experiment ID control and an Application name is used to run a protocol. When the object is created, the experimentID (string), applicationName (string), and iteration (int) properties of this object are accessible.

For details on the form design, see "Configuring a form for experiment IDs and Experiments database" on page 249.

Protocols run independently of a form

If your protocol contains Step Started/Completed tasks, but is designed to be run without a form, you must explicitly create your own protocolExperimentData object to add records to the Experiments database.

Properties of protocolExperimentData

protocolExperimentData has five properties:

- experimentID (string)
- applicationName (string)
- iteration (int)
- lastStartedStepName (string)
- stepCompleted (boolean)

The experiment ID, application name, and iteration number are used in combination to identify which run to associate with a given set of data in the Experiments database.

protocolExperimentData has no methods. You can use the protocolExperimentData object with the experimentTagsDB methods to make the required function calls. The following example shows how to declare your own protocolExperimentData object.

Example: Declaring protocolExperimentData object

```
var protocolExperimentData = {
experimentID: "expID1",
applicationName: "myApp",
iteration: 1,
lastStartedStepName: "",
stepCompleted: true
}
```

Methods of experimentTagsDB

Methods are JavaScript functions invoked through an object. The experimentTagsDB object has the following methods, available on systems that have a connection to the Experiments database. Use these methods to tag experiment data to be added to the Experiments database and to generate reports from the database.

Syntax:

experimentTagsDB.*method( )*

Note: Because a closed experiment cannot have data added to it, any calls to the following methods will fail if the experiment ID is Closed:

experimentTagsDB.CreateIteration();

experimentTagsDB.AddTag();

experimentTagsDB.AddTagToCurrentStep();

experimentTagsDB.AddStepCompletion();

experimentTagsDB.AddNote();

Note: Script errors are reported in the main Log and prevent the rest of the script from executing, but do not result in a VWorks Error dialog.

Method	Description

AddNote()	Adds a note to the specified experiment ID.
	Parameters:
	String. The text of the Note description.
	IMPORTANT This method requires that a protocolExperimentData object exist in the same JavaScript context, and that protocolExperimentData contains all of the standard data members created for it when a protocol is run via a form that uses the Experiments database. If protocolExperimentData does not exist, or if any of the fields are missing or incorrect, a call to experimentTagsDB.AddNote() will result in a script error.
	Example:
	Note: If your protocol is designed to be run without a form, you must explicitly create your own protocolExperimentData object. See Example: Declaring protocolExperimentData object.
	`var protocolExperimentData = {`
	`experimentID: "Experiment XYZ",`
	`applicationName: "Affinity Purification",`
	`iteration: 1,`
	`lastStartedStepName: "",`
	`stepCompleted: true`
	`}`
	`experimentTagsDB.AddNote("Off deck incubation");`

Method	Description

AddTagToCurrentStep()	Adds an experiment tag to the Experiments database as does calling `experimentTagsDB.AddTag()`. Except that `AddTag()` always specifies an order value of 1, whereas `AddTagToCurrentStep()` specifies whatever order value is supplied as the third parameter.
	Returns a Boolean value. It returns true if successful, and false if not successful.
	IMPORTANT This method requires that a protocolExperimentData object exist in the same JavaScript context and contain all of the standard data members created for it when a protocol is run via a form that uses the Experiments database. If protocolExperimentData does not exist, or if any of the fields are missing or incorrect, a call to experimentTagsDB.AddTagToCurrentStep() will result in a script error.
	Parameters:
	• *String*. The Field Name to be added to the database as a tag.
	This string must not be empty, otherwise, a script error will result.
	• *String*. The Field Value to be added with the tag.
	This field may be blank.
	• *Integer*. The order that the tag is to appear in its section of an experiment ID report.
	This value must be 1 or greater. If it is 0 or less, a script error results.
	Examples:
	Note: If your protocol is designed to be run without a form, you must explicitly create your own protocolExperimentData object. See Example: Declaring protocolExperimentData object.
	`var iter = experimentTagsDB.CreateIteration (experimentID, "Affinity Purification");`
	`experimentTagsDB.AddTagToCurrentStep("Volume uL", "100", 3)`
	`experimentTagsDB.AddTag(protocolExperimentData.experimentID, protocolExperimentData.applicationName, protocolExperimentData.iteration, "", "Custom Field 1", "JS inserted");`

CreateIteration()	Returns a long integer corresponding to an iteration number. Typically, AddTag() and CreateIteration() would be called at the start of the protocol, for example, in the startup script or in a task in the Startup Protocol. AddTag() would be called once for each tag to be created in the Experiments database. CreateIteration() would only be called once per protocol. Parameters: • *Text string*. The experiment ID. • *Text string*. The application name.

Method	Description
GetHighestIteration()	Returns a long integer corresponding to an iteration number. A return of zero means this application has not yet been run for this experiment ID. Parameters: • *Text string*. The experiment ID. • *Text string*. The application name.
GenerateReport	Generates a report (.pdf) of the specified experiment ID. Parameters: • *Text string*. The experiment ID. • *Text string*. The file path to the pdf file to be created. This second parameter can be blank, or omitted entirely. In this case, when this method call executes, a Save As file dialog box opens and prompts the user to specify the file name and storage location for the report. Example: `experimentTagsDB.GenerateReport (experimentID, "C:\VWorks Workspace\VWorks\ExperimentReports");`

Method	Description
StartOrCompleteStep()	Indicates the start and completion of a task sequence (step) to be tagged in the protocol for inclusion in the Experiments database. Returns a Boolean value: true if successful, and false if not successful. Parameters: • *String*. The name of the step to mark as started or completed. • *Boolean true/false*. Marks the step as started (false) or completed (true). If set to false (starting a step): – If the specified step has already been marked as started in the database, a warning will be displayed in the log, no action will be taken, and the function will return false. – Otherwise, the step will be marked as started, `protocolExperimentData.lastStartedStepName` will be set to the step specified in the first parameter, `protocolExperimentData.stepCompleted` will be set to false, and the function will return true. If set to true (completing a step): – If the specified step has not yet been marked as started or has already been marked as completed in the database, a warning will be displayed in the log, no action will be taken, and the function will return false. – Otherwise, the step will be marked as started, `protocolExperimentData.lastStartedStepName` will be set to the step specified in the first parameter, `protocolExperimentData.stepCompleted` will be set to true, and the function will return true. **IMPORTANT** This method requires that a `protocolExperimentData` object exist in the same JavaScript context and contain all of the standard data members created for it when a protocol is run via a form that uses the Experiments database. If `protocolExperimentData` does not exist, or if any of the fields are missing or incorrect, a call to `experimentTagsDB.StartOrCompleteStep()` will result in a script error. This method makes the same checks, and has the same effect, as executing a Step Started/Completed task. In the protocol, you would add a JavaScript task () with a call to this method at the point just before the start of the step to be tagged. You would add a second JavaScript task just after the end of this step. The parameters would be the same for both except for the final Boolean, which would be set to true, indicating that the step had completed. (A false value means that the step had started). Examples: *Note:* If your protocol is designed to be run without a form, you must explicitly create your own protocolExperimentData object. See Example: Declaring protocolExperimentData object. `experimentTagsDB.StartOrCompleteStep("Load Samples", false);`

Related information

For information about...	See...
Establishing connection to the experiment tags database Experiment ID reports	*VWorks Automation Control Setup Guide*
Using a form to track experiment data	"Configuring a form for experiment IDs and Experiments database" on page 249
Adding the Step Started/Completed task to a protocol	"Configuring a protocol to track experiment data" on page 260
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

InventoryLabware object

About the InventoryLabware object

The VWorks JavaScript interpreter provides an InventoryLabware object that can be accessed by a script. *InventoryLabware* represents an individual labware in the inventory database. By passing InventoryLabware objects to plateDB methods in a JavaScript, you can manage specific labware in the inventory. To see an example of this implementation, see the importLabwareToInventory method example in "plateDB object" on page 140.

Properties

Syntax:

InventoryLabware.*property*

Property	Data type	Description
name	String	Name of the labware.
device	String	Name of the device. **IMPORTANT** The device name must match the name in the device file. Be sure to include any spaces, underscores, dashes, and capitalization in the name.
cassette	Integer	Cassette number.
slot	Integer	Slot number.
type	String	Name of labware entry in the Labware Editor.
westbc	String	Barcode on the west side of the labware.
eastbc	String	Barcode on the east side of the labware.
southbc	String	Barcode on the south side of the labware.
northbc	String	Barcode on the north side of the labware.
status	String	Status of the labware: 0—OK 1—Error 2—Invalid 3—Barcode misread You can set or get the status of the labware. When you add the labware information to the database using the importLabwareToInventory plateDB method, you can set the labware status. When you retrieve labware information using the getLocationByBarcode plateDB method, you can get the labware status.

Property	Data type	Description
plategroup	String	Plate group to which the labware belongs.
locationgroup	String	Location group to which the labware belongs.

Methods

Methods are JavaScript functions invoked through an object. The InventoryLabware object has the following methods, available on systems that have the inventory database option. Use these methods to track labware-specific data.

Syntax:

InventoryLabware.*method( )*

Method	Description
nextPlateGroupName( )	Returns the name of the next plate group. A labware can belong to multiple plate groups. Use this method to determine all the plate groups to which a labware belongs.
nextLocationGroupName( )	Returns the name of the next location group. A labware can belong to multiple location groups. Use this method to determine all the locations groups to which a labware belongs.

Related information

For information about...	See...
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103
Other VWorks-defined objects	• "plate object" on page 120 • "runset object" on page 148 • "plateDB object" on page 140 • "task object" on page 126 • "forms object" on page 151
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

plate object

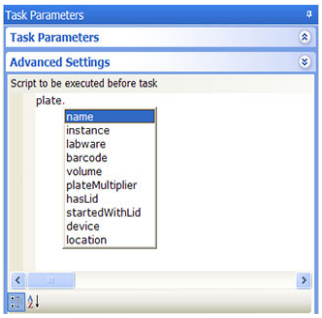
The VWorks JavaScript interpreter provides a plate object that can be accessed by a script. *plate* provides access to properties of the labware that the current task is operating on. You use the plate object in non-subprocess tasks such as the Place Plate task.

IMPORTANT To access labware properties in a subprocess task, use the plates[ ] object.

Properties

To see the properties of a plate object:

1. Open a protocol and select a task in the protocol area.
2. In the **Advanced Settings** area, type `plate.` The software automatically displays the list of available properties for the plate object.



Instead of displaying the list of plate properties in the Advanced Settings area, you can also list them in the Main Log. Doing so allows you to print the log file and retain a copy of the properties for reference.

To list the plate properties in the Main Log:

1. Open a protocol.
2. Select a task in the **Protocol** area.
3. In the **Task Parameters** area, click **Advanced Settings.**
4. Type the following in the **Script to be executed before task** area:

```
for (x in plate) {
   print ("plate." + x + " = " + plate[x]);
}
```

5. Run the protocol. The plate properties appear in the Main Log.

The following table lists the plate properties.

Property	Data type	Description
plate.name	Read-only string	Name of the process.
plate.instance	Read-only integer	Instance number of the process.

Property	Data type	Description
plate.labware	Read-only string	Name of the labware type.
plate.barcode	Read-only array	Array of four strings where, SOUTH=0, WEST=1, NORTH=2, EAST=3. The barcode property can be changed by the plate.setBarcode() method. Example: `plate.barcode[SOUTH]`
plate.volume	Read-only array of float arrays	An array consisting of one or more arrays of floating-point numbers. The array size depends on the number of wells in the labware (96, 384, or 1536), arranged in row, column format. This property is available only in VWorks software that has the volume-tracking database option. Example: `plate.volume[row][column]` For 96-well microplates, the maximum row value is 8, and the maximum column value is 12. For 384-well microplates, the maximum row value is 16, the maximum column value is 24. For 1536-well microplates, the maximum row value is 32, the maximum column value is 48.
plate.plateMultiplier	Read-only integer	The number of microplates needed for the run. This number is based on the number of times the protocol is run and the specifications in the Loop and Change Instance tasks. Examples: • A simple protocol processing Plate_A is run twice. The plateMultiplier is 2. • A protocol loops four times to move contents from the Source Plate to the Destination Plate, and changes instance on the Source Plate only. The protocol is run once. The plateMultiplier is 4 for the Source Plate and 1 for the Destination Plate.
plate.hasLid	Read-only boolean	The indicator that the microplate currently has a lid.

Property	Data type	Description
plate.startedWithLid	Read-only boolean	The indicator that the microplate entered the system with a lid.
plate.device	Read-only string	The device that is processing the microplate.
plate.location	Read-only string	The location on the device where the microplate currently resides.

Methods

Methods are JavaScript functions invoked through an object. The plate object has the following methods, available on systems that have the volume-tracking database option. Use these methods to track microplate-specific data.

Note: You can use the plate.getUserData() method to retrieve information stored by the plates[n].setUserData() method.

Method	Description
plate.setUserData(string *key*, string *value*)	Stores value under the key in a database record associated with this microplate. The data is retained across runs. The method is only available if the database is installed.
plate.getUserData(string *key*)	Returns the 'value' stored earlier using plate.setUserData or plates[n].setUserData. (See "plates[ ] object" on page 123.) The method is only available if the database is installed.
plate.reportErrorToPlugin(string *error*)	Calls the VWorks Hooks ScriptPlateError method to pass error to the Hooks plugin.
plate.setBarcode(SIDE, string *barcode*)	Manually stores the barcode data in plate.barcode[SIDE]. For example: `plate.setBarcode(WEST, "A1234")`

Related information

For information about...	See...
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103

For information about...	See...
Other VWorks-defined objects	• "InventoryLabware object" on page 118 • "runset object" on page 148 • "plates[ ] object" on page 123 • "plateDB object" on page 140 • "task object" on page 126 • "forms object" on page 151
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

plates[ ] object

The VWorks JavaScript interpreter provides a plates[ ] object that can be accessed by a script. *plates[ ]* is a VWorks-defined array of plate objects that provides access to properties of one or more microplates upon which the current task is operating. You use the plates[ ] object only for subprocess tasks that reference at least one microplate. For subprocess tasks that do not employ microplates, such as the curly-brace task, the plates[ ] object is not relevant.

IMPORTANT To access microplate properties in a non-subprocess task, use the plate object.

In a subprocess, the plates[ ] object is defined only for those tasks that employ a microplate, such as Aspirate or Dispense. Each element in the plates[ ] object references a single microplate that is referenced by the current instance of the task. The array elements refer to actual process plates, not just instances of a plate. The first element of the plates[ ] object is the first plate associated with that subprocess task. Therefore, plates[0] in the Aspirate task refers to the Source plate, while in the Dispense task, plates[0] is the Destination plate.

The array elements are in the order that they appear in the task. For example, the Dilute to Final Volume task uses two microplates: Source (plate[0]) and Destination (plate[1]). Suppose the task is in a loop and a Change Instance task changes the Destination microplate each time through the loop. The Source microplate remains plates[0] and the Destination microplate remains plates[1] each time the Dilute to Final Volume task is performed.

Properties

IMPORTANT You can list plates[ ] properties only for a subprocess task that has at least one microplate associated with it.

You can list the plate properties of each plate[n]. For example, you can list the properties for plate[0] and plate[1] to identify and determine the differences between them.

To list the plates[ ] properties in the Main Log:

1. Open a protocol.
2. In the **Protocol** area, select a subprocess task that references at least one microplate.
3. In the **Task Parameters** area, click **Advanced Settings.**
4. Type the following in the **Script to be executed before task** area:

```
for (x in plates[n])
{
   print ("plates[n]." + x + " =" + plates[n][x]);
}
```

   Where `n` is the array index.
5. Run the protocol. The plate properties appear in the Main Log.

   The plates[ ] properties are the same as the plate properties. For the list and description of the plate properties, see "plate object" on page 120.

Methods

Methods are JavaScript functions invoked through an object. The plates[ ] object has the following methods, available on those systems with the volume-tracking database option. Use these methods to track microplate-specific data.

Note: You can use the plates[n].getUserData() method to retrieve information stored by the plate.setUserData() method.

Method	Description
plates[n].setUserData(string *key*, string *value*)	Stores the *value* under the *key* in a database record associated with this microplate. The data is retained across runs. The method is only available if the database is installed.
plates[n].getUserData(string *key*)	Returns the *value* stored earlier using plates[n].setUserData or plate.setUserData. (See "plate object" on page 120.) The method is only available if the database is installed.
plates[n].reportErrorToPlugin(string *error*)	Calls the VWorks Hooks ScriptPlateError method to pass *error* to the Hooks plugin.
plates[n].setBarcode(SIDE, string *barcode*)	Manually stores the barcode data in plates[n].barcode[SIDE]. For example: `plates[0].setBarcode(WEST, "A1234")`

Related information

For information about...	See...
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103
Other VWorks-defined objects	• "InventoryLabware object" on page 118 • "plate object" on page 120 • "runset object" on page 148 • "plateDB object" on page 140 • "task object" on page 126 • "forms object" on page 151
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

task object

The VWorks JavaScript interpreter provides a task object that can be accessed by a script. *task* refers to the currently running task. It allows the properties of the task to be accessed using a standard syntax. Depending on which task is running, a different set of properties might be available.

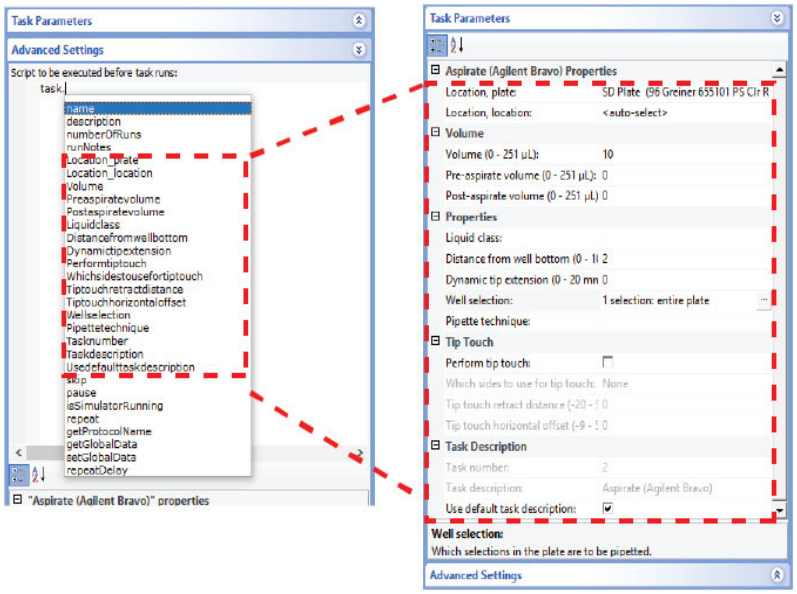
Properties

The properties available for a task correspond to its task parameters in the Task Parameters area.

To see the properties of a task:

1. Open a protocol that contains the task.
2. Select the task in the **Protocol** area.
3. In the **Advanced Settings** area, type `task.` in the **Script to be executed before task** box. The software automatically displays the list of available properties for the task object.

   In the following example, the properties for the Aspirate (Agilent Bravo) task is shown in the Advanced Settings area. Notice that they correspond to the parameters in the Task Parameters area.



Instead of displaying the list of task properties in the Advanced Settings area, you can also list them in the Main Log. Doing so allows you to print the log file and retain a hardcopy of the properties for reference.

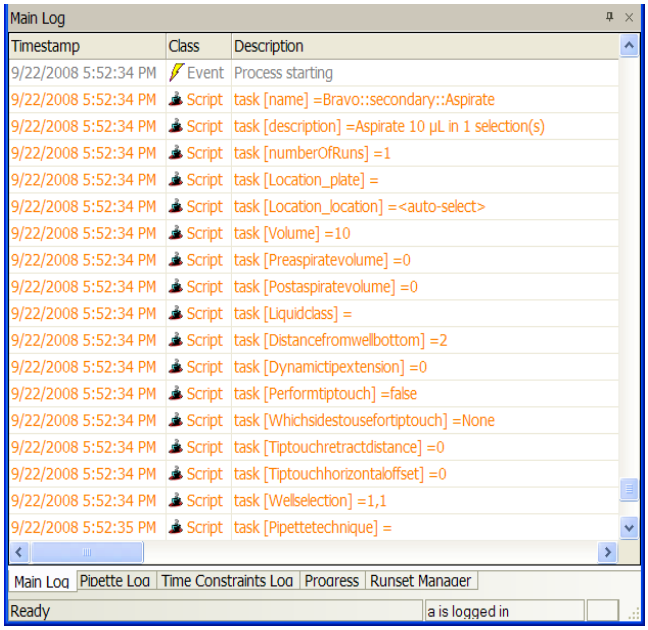
To list the task properties in the Main Log:

1. Open a protocol that contains the task.
2. Select the task in the **Protocol** area.
3. In the **Task Parameters** area, click **Advanced Settings.**
4. Type the following in the **Script to be executed before task** area:

```
for (x in task) {
   print("task.[" + x + "] =" + task[x]);
}
```

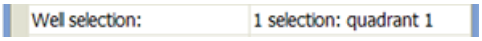
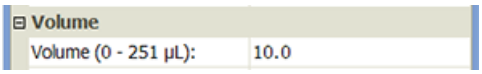
5 Run the protocol. The task properties appear in the Main Log.

The following example shows the properties for the Aspirate (Agilent Bravo) task.



Task property data types

A task property can be one of the following data types. You can use the default task parameter value to determine the data type.

Data type	Example task parameter
Array	Aspirate task, well selection  For more information about the array data type, see "Array data types" on page 128.
Boolean	Aspirate task, Perform tip touch option 
Float	Aspirate task, Volume parameter 
Integer	Loop task, Number of times to loop parameter 

Data type	Example task parameter
String	Name of a process plate 

Array data types

In JavaScript, an array is a built-in object that stores a collection of like values called elements. Each element is accessed by an index value that is enclosed in square brackets. Index values must be non-negative integers.

The following example script declares an array with three elements:

```
var vehicle_type = new Array(3);
vehicle_type[0] = "car";
vehicle_type[1] = "truck";
vehicle_type[2] = "van";
```

In the VWorks software, the array data type applies to properties that specify:

- The sides of a microplate for barcode reading or labeling
- The well selection in liquid-handling tasks, such as the Aspirate task

Sides of a microplate

In properties that specify the sides of a microplate, the array contains four elements, each representing a side of the microplate, for example:

- task.side[SOUTH]
- task.side[EAST]
- task.side[NORTH]
- task.side[WEST]

Well selection

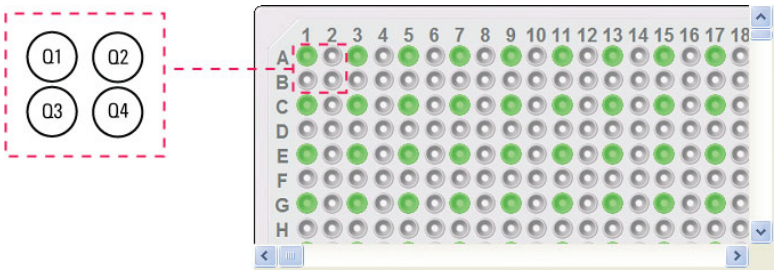
In properties that specify well selection, the array identifies the location of wells and quadrants in a microplate. Well locations are represented by a pair of integers that describe its row and column coordinates: [row, column].

A quadrant is an evenly spaced array of locations that are accessible by the tips on a pipette head. The following table lists the types of pipette heads and the number of accessible quadrants in various microplates.

Pipette head channels/ pin tool pins	Microplate	Number of quadrants
96	96-well	1
	384-well	4
	1536-well	16

Pipette head channels/ pin tool pins	Microplate	Number of quadrants
384	384-well	1
	1536-well	4
1536 (pin tool only)	1536-well	1

The following diagram demonstrates the concept of quadrants. The diagram shows a portion of a 384-well microplate and highlights the four quadrants (Q1, Q2, Q3, and Q4) that are accessible by the A1 tip of a 96-channel pipette head. Notice that the green color highlights all of the quadrant 1 (Q1) wells across the microplate.



When specifying well selection, you provide both the well location and quadrant location. For example, the designation for the four quadrants highlighted in the previous diagram is:

```
[[1,1],[1,2],[2,1],[2,2]]
```

where the Q1 coordinates are [1,1], Q2 coordinates are [1,2], Q3 coordinates are [2,1], and Q4 coordinates are [2,2].

You can also use variables to represent the row and column coordinates, as the following example shows:

```
task.Wellselection = [[disp_row,disp_col]]
```

IMPORTANT Quadrant specifications in the Well Selection dialog box will override task.Wellselection values assigned in the Advanced Settings area.

To cycle through the quadrants in a liquid-handling task, you can use a series of *if* statements as the following example shows.

```
var Dispense_Loop;
if (Dispense_Loop == undefined) {
      Dispense_Loop = 1;
}
if (isNaN(Dispense_Loop)){
   Dispense_Loop = 1;
}
if ((Dispense_Loop < 1)||(Dispense_Loop > 4)){
   Dispense_Loop = 1;
}
```

```
if (Dispense_Loop == 1){
   task.Wellselection = [[1,1]];
}
if (Dispense_Loop == 2){
   task.Wellselection = [[1,2]];
}
if (Dispense_Loop == 3){
   task.Wellselection = [[2,1]];
}
if (Dispense_Loop == 4){
   task.Wellselection = [[2,2]];
}
Dispense_Loop++;
print("Dispensing to quadrant "+task.Wellselection+" of
Destination plate.");
}
```

Methods

The task object methods, and the properties of the methods, are specific to the selected task. The following table lists some of the shared methods:

Method	Description

task.skip()	Skips the current task. Use this function to conditionally run a task, such as in this example which skips the task if the simulator is not running:
	`if (!task.isSimulatorRunning()) {`
	`task.skip();`
	`}`

task.pause()	Pauses the protocol and opens a dialog box that asks you whether you want to continue or abort the run. Use this function if you need to pause the protocol to, for example, replenish the fluid in a static reservoir. You could use the print() function to add a note to the log toolbar describing the action to take when the VWorks software has paused.
task.isSimulatorRunning()	Returns true if this is a simulated run. Has no arguments.

Method	Description
task.repeat()	Schedules another instance of this task, including its script, to be repeated immediately after this task finishes execution. Subsequent calls to task.repeat() within the same script instance are ignored.
task.repeatDelay(int *timeseconds*)	Requests the task to wait the specified time (in seconds), and then repeats. The clock starts running the moment this call is made, but the subsequent instance of this task will not start executing before the current instance finishes execution. If task.repeatDelay() is called more than once within the same script instance, the final call's specified number of seconds will be used. In no case will task.repeatDelay() cause more than one additional instance of this task to be scheduled.
task.setGlobalData(string *name*, string *value*)	Stores *value* in *name* in the Global database.
task.getGlobalData(string *name*)	Returns the value stored earlier using task.setGlobalData for the given *name*.
task.getProtocolName()	Retrieves the full protocol path (string).

task.Headmode method

Available only for SubProcess (Bravo). The following method enables selection of a subset of the barrels in the pipette head:

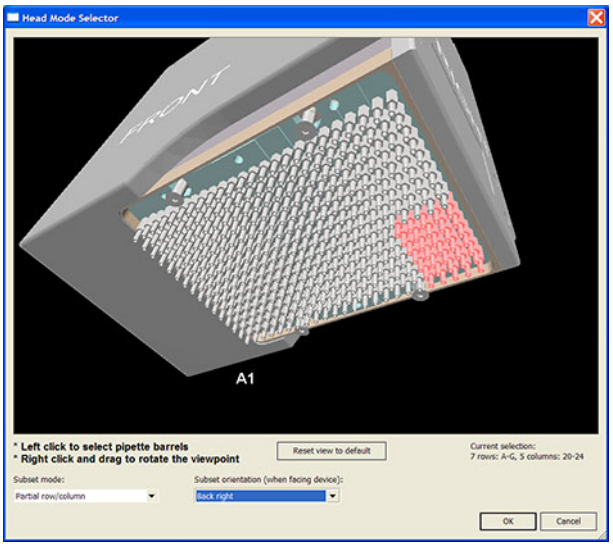
task.Headmode(string *args*)

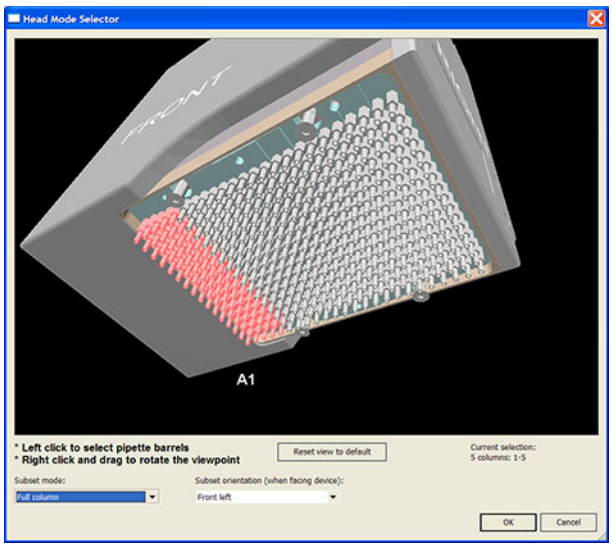
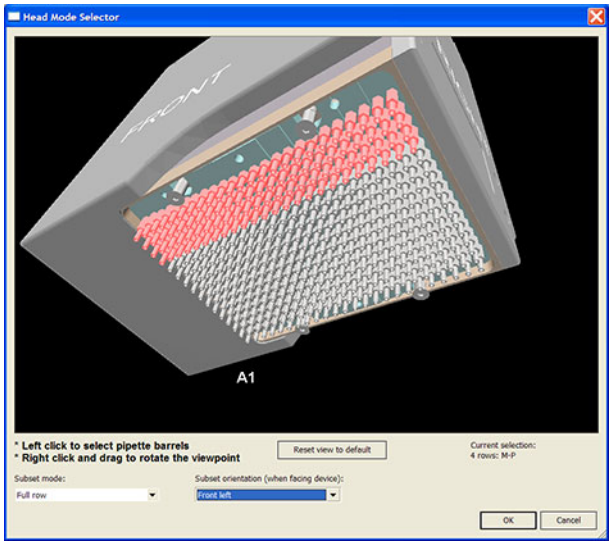
The string parameter must contain a comma-separated list of the following four interger values:

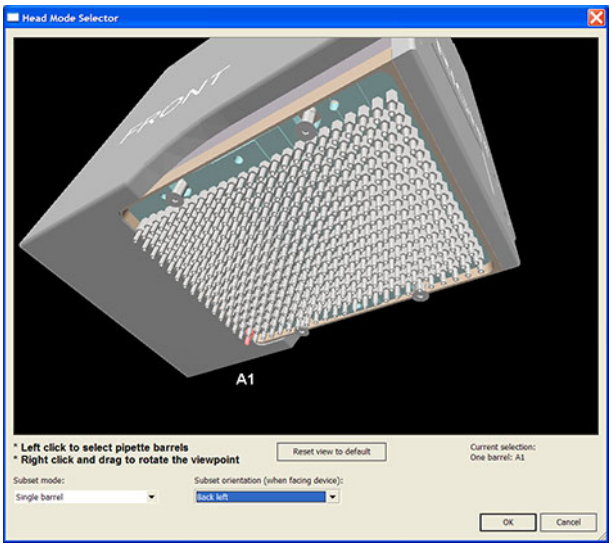
string args	Values
1	Specifies the barrel selection as follows: 0 = all_barrels 1 = column 2 = row 3 = partial_column_row 4 = single_barrel

string args	Values
2	Specifies the region of the barrel selection, where, 0 = front_right 1 = back_right 2 = back_left 3 = front_left
3	Specifies the number of rows for the barrel selection, for example, • 96-barrel head, the maximum row value is 8, and the maximum column value is 12. • 384-barrel head, the maximum row value is 16, the maximum column value is 24.
4	Specifies the number of columns for the barrel selection, for example, • 96-barrel head, the maximum row value is 8, and the maximum column value is 12. • 384-barrel head, the maximum row value is 16, the maximum column value is 24.

The following examples show how to use task.Headmode.

Barrel selection options	Example task.Headmode script and comparable view in the Head Mode Selector dialog box
Select partial row/ column	Selecting 35 barrels (7-by-5 region) `task.Headmode = "3,1,7,5"` 

Barrel selection options	Example task.Headmode script and comparable view in the Head Mode Selector dialog box
Select column or columns	Selecting 5 columns *Note:* Row specified is ignored. `task.Headmode = "1,2,1,5"` 
Select row or rows	Selecting 4 rows *Note:* Column specified is ignored. `task.HeadMode = "2,3,4,1"` 

Barrel selection options	Example task.Headmode script and comparable view in the Head Mode Selector dialog box
Select single barrel	Selecting single barrel in the back left *Note:* Row and column specified are ignored; region is always 1-by-1. `task.HeadMode = "4,2,1,1"` 

Related information

For information about...	See...
Writing JavaScript for the Print and Apply task	"Using JavaScript with the CentrifugeAuto task" on page 136
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103
Other VWorks-defined objects	• "InventoryLabware object" on page 118 • "plate object" on page 120 • "plates[ ] object" on page 123 • "plateDB object" on page 140 • "runset object" on page 148 • "forms object" on page 151
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

For information about...	See...
Specifying a quadrant pattern in the Well Selection dialog box	• "Aspirate (Agilent Bravo)" on page 455 • "Dispense (Agilent Bravo)" on page 514 • "Mix (Agilent Bravo)" on page 559

Using JavaScript with the CentrifugeAuto task

This topic provides two examples of using JavaScript to modify the implementation of the CentrifugeAuto task. For more information about using the CentrifugeAuto task, see "CentrifugeAuto" on page 353.

Counterweight selection

When using the Auto managed counterweight mode, you can use JavaScript to select the counterweight, for example:

```
if(place.instance%2 == 1)
{
task.Counterweight = "CW1";
}
else
{
task.Counterweight = "CW2";
}
```

Reduce number of spins for one plate

When running an odd number of microplates with the CentrifugeAuto task in Use 2 protocol plates mode, the second to last plate is spun twice so that the last microplate is paired. Using JavaScript, you can direct the protocol to switch to Auto managed counterweight mode for the last spin, for example:

```
if((plate.instance == task.numberOfRuns) &&
(plate.instance%2 == 1))
{
task.skip();
}
if((plate.instance == task.numberOfRuns) &&
(plate.instance%2 ==1))
{
task.Counterweight = "CW1"
}
else
{
task.skip();
}
```

Related information

For information about...	See...
CentrfiugeAuto task	"CentrifugeAuto" on page 353
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103
Using JavaScript utilities	"Using JavaScript utilities" on page 153

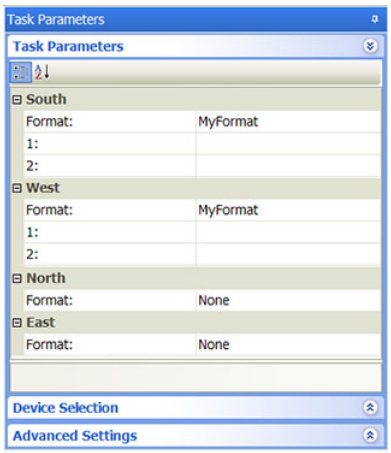
About scripting the Print and Apply task

Print and Apply task parameter exceptions

Unlike other tasks, the Print and Apply task parameters have the following exceptions:

- The number of parameters can vary, because each labeling device has a different number of formats, and each format has a different number of fields. (In the software, the contents of each field are specified in the Field Composer dialog box.)
- The software uses the destination side of the microplate (south, west, north, or east) as a prefix for the parameter names:

  In the following example, MyFormat is specified for the south and west sides of a microplate for fields 1 and 2. The software will append a prefix to each field name to indicate the side. The new field names will be South_1, South_2, West_1, and West_2.



Scripting what to print (label content)

When scripting the task parameters for the Print and Apply task, the field parameter must be assigned to a string that encodes what is to be printed. The following table lists the task parameters and the corresponding string options.

Note: When in doubt about available task parameters, add the following script into the task and run: `for(x in task){print(x + "::" + task[x])}` This script prints all the current task parameters.

Parameter	JavaScript string

Date	`"[DATE:enum1]"`
	Where `enum1`:
	0 – Use System Format
	1 – YYYY/MM/DD
	2 – DD/MM/YYYY
	3 – MM/DD/YYYY
	4 – YY/MM/DD
	5 – DD/MM/YY
	6 – MM/DD/YY
	Example:
	`task.East_1 = "[DATE:0]"`

Time	`"[TIME:enum1]"`
	Where `enum1`:
	0–12 (12-hour time AM/PM)
	1–24 (24-hour time)
	Example:
	`task.East_1 = "[TIME:0]"`

Counter	`"[COUNTER:StartAt:IncBy:TotalNumberOfDigits:IncEveryNPlates:enum1]"`
	Where `enum1`:
	0 – Numeric (0–9)
	1 – Alphanumeric (0–Z)
	Example:
	`task.East_1 = "[COUNTER:0:1:3:2:1]"`

File With Lookup	`"[FILEWLOOKUP:filename:keyCol:ValueCol]"`
	Example:
	`task.East_1 = "[FILEWLOOKUP:c:/temp.csv:1:1]"`

File Start at row/col and increment row	`"[FILE:filename:startRow:startCol]"`
	Example:
	`task.East_1 = "[FILE:c:/temp.csv:1:1]"`

Parameter	JavaScript string

Static Text	`"[STATIC:staticText]"`
	Example:
	`var my_string = "hello world";`
	`task.East_1 = "[STATIC:"+ my_string  +"]";`

CopyBarcode from a different side	`task.East_1 = "[BARCODE:copyFromSideEnum1]"`
	Where `copyFromSideEnum1`:
	0 – south
	1 – west
	2 – north
	3 – east
	Example:
	`task.East_1 = "[BARCODE:2]"`

Guidelines for scripting where to print the barcode label

About scripting which field to print to

To print on a single side or more than one side, use the following syntax:

```
task.East_1
```

where, `East` is the side of the labware and `1` is field 1.

About scripting the format

Although it is possible to script the format, you must ensure that the scripted format has the same number of fields as the format specified in the protocol.

```
task.West_Format=3
//script the format
task.West_3 = "[FILEWLOOKUP:C:\\bcf.csv:1:3]"
//script field 3 of format 3
```

Note: To read and write the field task parameters, the values used are pre-evaluation. For example, to print the current time, you would set the task parameter for field 1 to `[TIME:0]`. If you print this from JavaScript using `print(task.East_1)`, the result will be `[TIME:0]`, not as the actual time.

Related information

For information about...	See...
Print and Apply task	"Print and Apply" on page 381
Label formats	*Microplate Labeler User Guide*
Field contents	"Print and Apply" on page 381
Using JavaScript in the VWorks software	"Using JavaScript" on page 100

plateDB object

The VWorks JavaScript interpreter provides a plateDB object that can be accessed by a script. *plateDB* enables control of the inventory database through JavaScript. You can use the plateDB object to create plate groups, add plates to existing plate groups (if they exist in a location group), remove plates from plate groups, and delete plate groups.

Properties

The plateDB object has no properties.

Methods

Methods are JavaScript functions invoked through an object. The plateDB object has the following methods, which are available on systems with the inventory database option.

Syntax:

plateDB.*method( )*

Note: A common set of return codes are used in the applicable methods. However, based on relevance, an applicable method might use only a subset of the return codes.

Method	Description
createPlateGroup(string *plateGroup*)	Creates a new plate group. Returns true on success, or returns false on failure. The method also prints to the log the reason for failure. Example: `plateDB.createPlateGroup("myplategroup")`
deletePlateGroup(string *plateGroup*)	Deletes a plate group. Returns true on success, or returns false on failure. The method also prints to the log the reason for failure. Example: `plateDB.deletePlateGroup("myplategroup")`
addPlateToGroup(int *side*, string *barcode*, string *locationGroup*, string *plateGroup*)	Adds a labware that exists in a location group to a plate group. Returns true if successful, or returns false upon failure. The method also prints the reason for failure to the log. *Note:* The plate must exist in a locationGroup, and the given barcode must be on the specified side of the plate. Example: `plateDB.addPlateToGroup(WEST,"BC01","locationGroup","plateGroup")`
deleteAllPlatesFromGroup(string *plateGroup*)	Deletes all labware from a specified plate group. Returns false upon failure, and prints the reason for failure to the log. *Note:* The plate group must exist. Example: `plateDB.deleteAllPlatesFromGroup("plateGroupName")`

Method	Description

importCsvToInventory(string *CSVFilepath*, bool *OverwriteInventory*)	Imports the contents of the specified CSV file into the inventory database.
	You have the option of overwriting the existing data in the database during the import process:
	true—Overwrite the existing data.
	false—Skip the current labware.
	The method returns the following:
	–1—Unknown error.
	0—Method is successful.
	1—Location is occupied by another labware.
	2—Location is empty.
	3—Device does not exist.
	4—Cassette does not exist.
	5—Slot does not exist.
	6—CSV file does not exist in the specified directory.
	Example (local file system):
	`plateDB.importCsvToInventory ("C:\\VWorks Workspace\\labware.csv", true)`
	Example (Shared Services storage):
	`plateDB.importCsvToInventory ("[olssvr]:VWorks Projects/VWorks/Miscellaneous/labware.csv", true)`

Method	Description

importLabwareToInventory(array *Labware*, bool *OverwriteInventory*)	Imports the array of labware data into the inventory database. The InventoryLabware object is passed to this method and is an element in the array.
	Each array element (labware) must have the following information:
	• Device
	• Cassette number
	• Slot number
	• Labware type
	You have the option of overwriting the existing data in the database during the import process:
	true—Overwrite the existing data.
	false—Skip the current labware.
	The method returns the following:
	1—Error. For errors in importing the labware, the error description is printed to main log.
	0—Method is successful.
	Example:
	`var LabwareArray = new Array();`
	`var Labwareone = new InventoryLabware();`
	`Labwareone.device = "Agilent Labware MiniHubl";`
	`Labwareone.cassette = "1";`
	`Labwareone.slot = "1";`
	`Labwareone.type = "384 Greiner ";`
	`Labwareone.westbc = "west";`
	`LabwareArray[0] = Labwareone;`
	`plateDB.importLabwareToInventory(LabwareArray,true );`

clearInventory()	Deletes all labware data in the inventory database. The method returns the following: 1—Error. The error description is printed to the main log. 0—Method is successful. Example: `plateDB.clearInventory();`

Method	Description

clearStorageLocation(string *GroupName*)	Deletes labware in the specified location group or plate group.
	The method returns the following:
	1—Error. The error description is printed to the main log.
	0—Method is successful.
	Example:
	`plateDB.clearStorageLocation("MyPlateGroup");`
	`plateDB.clearStorageLocation("MyLocationGroup");`

clearAbandonedLabware( )	Deletes labware that has become abandoned. Labware becomes abandoned when it moves out of storage and is somewhere in the system when a deadlock occurs or when a run is aborted. The method returns the following: 1—Error. The error description is printed to the main log. 0—Method is successful. Example: `plateDB.clearAbandonedLabware();`
clearAllLabwareBarcodes( )	Deletes all labware barcodes in the inventory database. Other labware information is retained. The method returns the following: 1—Error. The error description is printed to the main log. 0—Method is successful. Example: `plateDB.clearAllLabwareBarcodes();`

Method	Description

addLabwareToLocation(string *DeviceName*, string *CassetteNumber*, string *SlotNumber*, string *Type*, bool *Overwrite*)	Adds a labware to the specified storage location in the inventory database.
	The method returns the following:
	0—Method is successful.
	3—Cassette or slot is set to 'all' and the device does not exist or is disabled.
	4—Cassette out of range.
	5—Slot out of range.
	1—Other error.
	The error description is printed to the main log.
	Example:
	`plateDB.addLabwareToLocation("Agilent Labware MiniHub","1","1","384 Greiner",true);`
	`plateDB.addLabwareToLocation("Agilent Labware MiniHub","all","1","384 Greiner",true);`
	`plateDB.addLabwareToLocation("Agilent Labware MiniHub","1","all","384 Greiner",true);`
	`plateDB.addLabwareToLocation("Agilent Labware MiniHub","all","all","384 Greiner",true);`

deleteLabwareAtLocation(string *DeviceName*, string *CassetteNumber*, string *SlotNumber*)	Deletes a labware from the specified storage location.
	The method returns the following:
	0—Method is successful.
	2—Location is empty.
	3—Cassette or slot is set to 'all' and the device does not exist or is disabled.
	4—Cassette out of range.
	5—Slot out of range.
	1—Other error.
	The error description is printed to the main log.
	Example:
	`plateDB.deleteLabwareAtLocation("Agilent Labware MiniHub","1","1");`
	`plateDB.deleteLabwareAtLocation("Agilent Labware MiniHub","all","1");`
	`plateDB.deleteLabwareAtLocation("Agilent Labware MiniHub","1","all");`
	`plateDB.deleteLabwareAtLocation("Agilent Labware MiniHub","all","all");`

Method	Description

getLocationByBarcode(string *Barcode*)	Obtains the location of a labware using the supplied barcode.
	Returns an array of labware (InventoryLabware objects) if the method is successful. Otherwise, the method returns a null array.
	Example:
	`var LabwareArray = new Array();`
	`LabwareArray = plateDB.getLocationByBarcode("barcode");`

exportDatabase(string *FileName*)	Exports the data in the inventory database to an SQL file.
	The method returns the following:
	0—Method is successful.
	6—File path is invalid.
	1—Other error.
	Example (local file system):
	`plateDB.exportDatabase("C:/VWorks Workspace/exportedfile.sql")`
	Example (Shared Services storage):
	`plateDB.exportDatabase("[olssvr]:VWorks Projects/VWorks/Miscellaneous/SQL Files/exportedfile.sql")`
	Note: The "[olssvr]:" prefix to the file path specifies the exported file as a record of Interest and opens the Audit Comment dialog box.

importDatabase(string *FileName*)	Imports the data from the specified SQL file into the inventory database.
	The method returns the following:
	0—Method is successful.
	6—File path is invalid.
	1—Other error.
	Example (local file system):
	`plateDB.importDatabase("C:/VWorks Workspace/importedfile.sql")`
	Example (Shared Services storage):
	`plateDB.importDatabase("[olssvr]:VWorks Projects/VWorks/Miscellaneous/SQL Files/importedfile.sql")`

enumeratePlateGroup(string *plateGroup*)	Returns an array of strings corresponding to the IDs of all plates in the specified plate group. If the method fails, the reason for failure prints to the log. *Note:* The plate group must exist. Example: `plateDB.enumeratePlateGroup("plateGroupName")` See additional example in getPlateBarcode method, below.

Method	Description

getPlateBarcode(int *side*, array *plateID*)	Returns the plate barcode (string) on the specified side of the specified plate.
	If the method fails, the reason for failure prints to the log.
	Example:
	`var plateIDs = plateDB.enumeratePlateGroup("plateGroupName")`
	`for(var plateID in plateIDs){`
	`print(plateDB.getPlateBarcode (WEST, plateIDs[plateID]))`
	`}`

enumerateListOfGroups(int *groupType*)	Returns an array of strings consisting of all group names in the Inventory database. *groupType* specifies which kind(s) of groups, where 0 = both types (plate groups and location groups) 1 = plate groups only 2 = location groups only If the method fails, the reason for failure prints to the log. Example: `var group_array = plateDB.enumerateListOfGroups(0)`
getLabwareInfo(string *labwareName*)	Returns a string containing well information about the given labware entry in the following format: "NumberOfWells=<number>" If the labware database does not contain a labware entry with the given name, the method returns an empty string. Example: `plateDB.getLabwareInfo("96 Greiner Polystyrene Clear Round Well, Flat Bottom")`

Related information

For information about...	See...
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103
Other VWorks-defined objects	• "InventoryLabware object" on page 118 • "plate object" on page 120 • "plates[ ] object" on page 123 • "runset object" on page 148 • "task object" on page 126 • "forms object" on page 151

For information about...	See...
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

runset object

The VWorks JavaScript interpreter provides a runset object that can be accessed by a script. *runset* enables control of the Runset Manager through JavaScript. You can use the runset object to clear the runs from a runset or append protocols to a runset. You can also query the Runset Manager and retrieve various fields of interest, such as barcodes. For more information on runsets, see "Managing runsets" on page 317.

Note: If you are using the VWorks Watcher, the runset object can be used in a script that is called by Watcher to create runsets automatically. For details, see "Setting up and using the Watcher tool" on page 755.

Properties

The runset object has no properties.

Methods

Methods are JavaScript functions invoked through an object. The runset object has the following methods.

Method	Description
clear()	Clears all of the entries in the runset manager, except for currently executing protocols, which are not affected. Example: `runset.clear()`
appendProtocolFileToRunset(string *ProtocolPath*, int *RunTimes*, string *ProtocolNotes*, string *formToUse*)	Appends the specified protocol file at *ProtocolPath* to the Runset Manager. Specifies the number of *RunTimes,* and provides any *ProtocolNotes*. To specify a VWorks form to be associated with the protocol, *formToUse* must specify the file name of the form without the path, for example, C:\MyDir\MyForm.VWForm must be specified as "MyForm.VWForm". Returns true if successful or false upon failure. If a failure occurs, for example, the .pro file is missing, errors are written to the log.
Example: `runset.appendProtocolFileToRunset("[olssvr]:VWorks Projects/VWorks/Protocols/General/someprotocol.pro",10, "","FormName1.VWForm");`	

Method	Description
appendProtocolFileToRunset AtTime(string *ProtocolPath*, int *RunTimes*, string *ProtocolNotes*, string *formToUse*, string *AtTime*)	Appends the specified protocol file at *ProtocolPath* to the Runset Manager at the date and time (local time) specified by *AtTime*. Specifies the number of *RunTimes*, provides any *ProtocolNotes* that will appear in the Runset Manager pane. To specify a VWorks form to be associated with the protocol, *formToUse* must specify the file name of the form without the path, for example, C:\MyDir\MyForm.VWForm must be specified as "MyForm.VWForm". Returns true if successful or false upon failure. If a failure occurs, for example, the protocol file (.pro) is missing, errors are written to the log.

Example:
`var d = new Date();`
`ProtocolStartTime  = d.toString();`
`runset.appendProtocolFileToRunsetAtTime("[olssvr]:VWorks Projects/VWorks/Protocols/General/Protocol File - 1.pro",10,"","FormName1.VWForm",ProtocolStartTime);`	

appendRunsetFileToRunset (string *runsetPath*, string *formToUse*)	Appends the contents of a runset file, located at *runsetPath*, to the Runset Manager. To specify a VWorks form to be associated with the protocol, *formToUse* must specify the file name of the form without the path, for example, C:\MyDir\MyForm.VWForm must be specified as "MyForm.VWForm". Returns true if successful or false upon failure. If a failure occurs, for example, the .rst file is missing, errors are written to the log.

Examples:
`runset.appendRunsetFileToRunset("[olssvr]:VWorks Projects/VWorks/Runsets/General/runset.rst");`
`runset.appendRunsetFileToRunset("[olssvr]:VWorks Projects/VWorks/Runsets/General/runset.rst", "FormName1.VWForm");`	

openRunsetFile(string *runsetPath*, string *formToUse*)	Opens the runset located at *runsetPath*. Optionally, opens the form to use to run the runset*formToUse* must specify the file name of the form without the path, for example, C:\MyDir\MyForm.VWForm must be specified as "MyForm.VWForm". Returns true if successful or false upon failure. If a failure occurs, for example, the .rst file is missing, errors are written to the log.

Method	Description

Examples:
`runset.openRunsetFile("[olssvr]:VWorks Projects/VWorks/Runsets/General/my_runset.rst");`
`runset.openRunsetFile("[olssvr]:VWorks Projects/VWorks/Runsets/General/my_runset.rst", "FormName1.VWForm");`	

getRunsetXML()	Returns the XML contents of a runset file (.rst).

Example for getRunsetXML()

```
// Note: In this example, runsetParser is a hypothetical
JavaScript function.
var runsetXML = runset.getRunsetXML()
var runsetParser = new runsetParser(runsetXML)
var numRunningProtocols =
runsetParser.GetNumberOfRunningProtocols()

for(  var running_protocol_idx  = 0;
      running_protocol_idx < numRunningProtocols;
      running_protocol_idx++){

   print(runsetParser.GetRunningProtocol(running_protocol_i
   dx).GetFilename())
   print(runsetParser.GetRunningProtocol(running_protocol_i
   dx).GetRunsetNotes())
   //The runset notes might include items such as barcodes.
}
```

Related information

For information about...	See...
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
VWorks-defined functions	"VWorks-defined functions" on page 103
Other VWorks-defined objects	• "InventoryLabware object" on page 118 • "plate object" on page 120 • "plates[ ] object" on page 123 • "plateDB object" on page 140 • "task object" on page 126 • "forms object" on page 151
Using JavaScript utilities	"Using JavaScript utilities" on page 153
JavaScript task	"JavaScript" on page 653

For information about...	See...
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75
VWorks Watcher tool	"Setting up and using the Watcher tool" on page 755
How to create a runset	"Managing runsets" on page 317

forms object

The VWorks JavaScript interpreter provides a forms object that can be accessed by a script. *forms* provides the same function as the Toggle Full Screen button, which is one of the specialized buttons that you can include on a form that runs a protocol. For more information on forms, see "Creating forms for protocol operators" on page 177.

Properties

The forms object has no properties.

Methods

Methods are JavaScript functions invoked through an object. The forms object has the following method.

Method	Description

setFullScreen(Boolean *True/False*)	Changes the display of the VWorks window between a full-screen view of a protocol form and a normal tabbed view.
	Example:
	`forms.setFullScreen(true)`
	`forms.setFullScreen(false)`

Related information

For information about...	See...
Designing forms to run protocols	"Creating forms for protocol operators" on page 177
VWorks-defined functions	"VWorks-defined functions" on page 103

Using start and finish protocol scripts

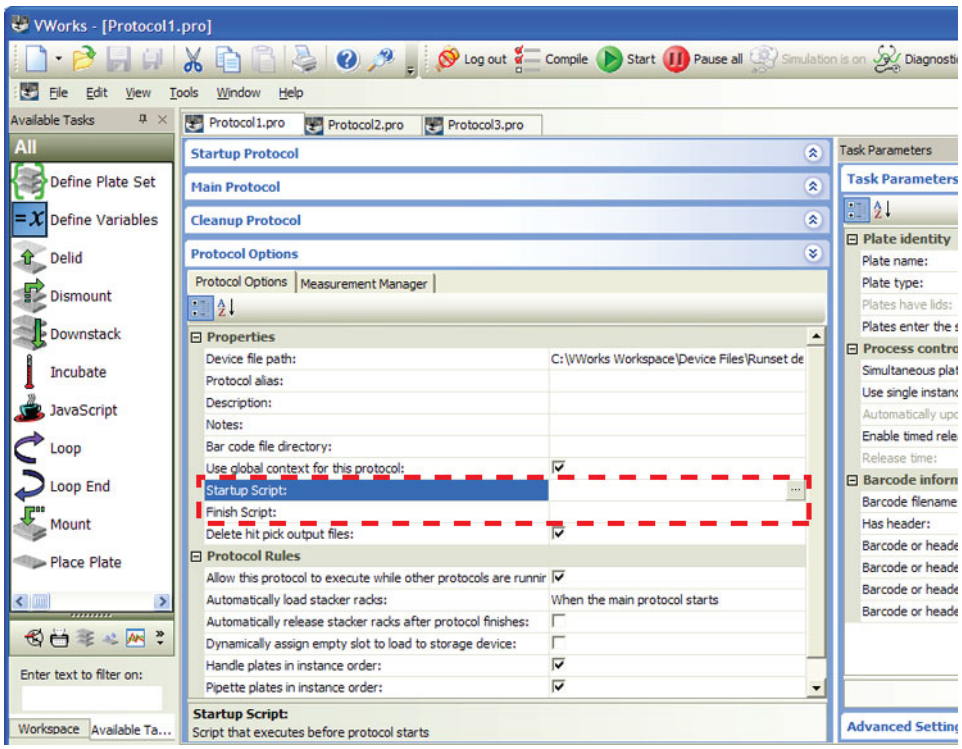
A startup protocol script is a JavaScript that runs before the Startup Protocol begins. A finish protocol script is a JavaScript that runs after the Cleanup Protocol finishes. For example, you can use an open () statement to open and load a file.

Procedure

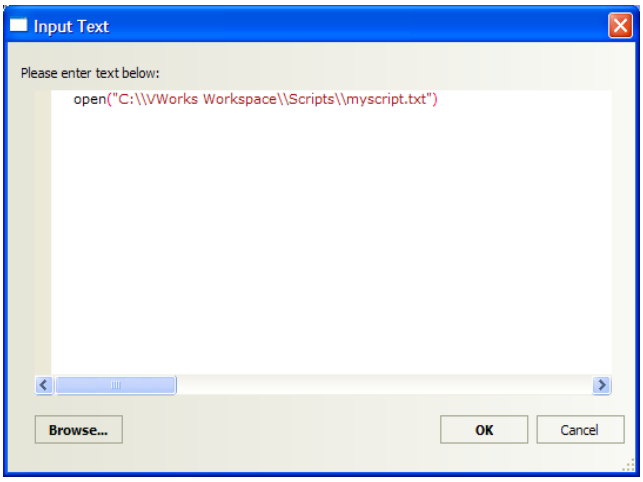
To add start or finish scripts:

1 Open the protocol file, and then click **Protocol Options.**

2 Click the **Startup Script** or the **Finish Script** field, and then click the ··· button that appears.



The Input Text dialog box appears.



3 Do one of the following:
   - Type the JavaScript code in the box.
   - Click **Browse**. In the **Open** dialog box, select a file that contains the JavaScript code, and click **Open**.
4 Click **OK** to save the changes and return to the VWorks window.

Related information

For information about...	See...
Opening protocols	"Opening a protocol" on page 291
Setting protocol options	"Setting protocol options" on page 42
Using script variables directly in task parameters	"Using simple variables" on page 93
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75

Using JavaScript utilities

Two JavaScript utilities are available:

- *ActiveX Wrapper*. Allows you to create an ActiveX object and use the associated ActiveX methods.
- *File Object*. Allows you to create a file object to read from and write to a file.

You can write JavaScript using these utilities in the VWorks Advanced Settings area. For instructions on how to add the JavaScript task and display the Advanced Settings area, see "Where to write JavaScript" on page 100.

ActiveX Wrapper

ActiveX controls are software components that allow different software products to interact. For example, if you want to use the VWorks software to control a device, you can use the device's ActiveX control to invoke the device's operations.

The ActiveX Wrapper utility in the VWorks software allows you to use another product's ActiveX control to invoke the product's operations. Make sure you install the product's ActiveX control software before you run the JavaScript.

To use the ActiveX utility, you need to:

1. Create an ActiveX object to reference the ActiveX control.
2. Call the associated ActiveX methods to invoke the ActiveX operations.
3. Use the Set and Get methods to access the ActiveX properties.

Create an ActiveX object

To create an ActiveX object:

In the Advanced Settings area, type the JavaScript code to create an ActiveX object.

For example, if the ActiveX control PROG_ID is PlateLocCtrl.2, you can create the object as follows:

```
var ocx
if(ocx==undefined){
ocx=new ActiveX("PLATELOC.PlateLocCtrl.2");
}
```

The var statement declares a JavaScript variable. In this example, the variable is `ocx`.

The if statement prevents the software from creating the ActiveX multiple times if the script is run repeatedly.

The `ocx = new ActiveX` statement passes the PlateLoc PROG_ID to the ActiveX object generator. Using the ID, the generator calls the CreateInstance API. The resulting ActiveX object is then wrapped in the scripting layer that translates arguments and returns values that are understood by both the PlateLoc Sealer and the VWorks software.

Calling the ActiveX methods

To call the ActiveX methods:

Call the methods using the following syntax:

```
objectname.method
```

For example, if you want to call the AboutBox() method, you can type the following:

```
ocx.AboutBox()
```

To list the available ActiveX methods, use the following JavaScript statements:

```
for(x in ocx.members)
print(x)
```

Accessing the ActiveX properties

To access the ActiveX properties:

Use the Set or Get method and the following syntax:

- comm.set("property_name", value)

- comm.get("property_name")

For example, if you want to access the ActiveConnection property of an ADO command object, you can type the following:

```
var db=new ActiveX("ADODB.Connection")
var comm=new ActiveX("ADODB.Command")
comm.set("ActiveConnection", db)
```

The first `var` statement creates an ADO object and assigns it to a variable named db.

The second `var` statement creates an ADO command object and assigns it to a variable named comm.

The `comm.set` statement sets the ActiveConnection property to the connection object in the db variable.

File Object

The File Object utility allows you to create a file object so that you can read from and write to a file. The file can be in local file system storage or Shared Services storage (OLSS).

To use the File Object utility:

1. Create the file object.
2. Call the desired file object methods:
   - Open()
   - Close()
   - Read()
   - Write()
   - IsOpen()
   - Exists()
   - Delete()

   Note: The Delete() and Write() methods are not available for files in OLSS storage.

IMPORTANT

The JavaScript language is case-sensitive. Make sure you use the correct upper- and lower-case letters when calling the methods.

Creating a file object

To create a file object:

In the Advanced Settings area, type the following JavaScript code:

```
var fileobjectname
if(fileobjectname==undefined){
fileobjectname=new File()
}
```

Note: fileobjectname is the name of the file object you want to create.

Calling the Open() method

To call the Open() method:

Type the following JavaScript code:

```
fileobjectname.Open("filepath",0,0)
```

`fileobjectname` is the name of the file object you created.

`filepath` (the first argument) is the location of the file you are creating. For example, you can type `c:\\fileobjectname.txt`.

`0` (the second argument) specifies how new information will be added to the file. 0 adds new information after the existing information. A non-zero value erases the existing file contents before adding the new information. If you do not specify this argument, the system will use the default value of 0.

`0` (the third argument) specifies how the line endings in binary files will be translated. 0 translates line endings to a carriage return followed by a line feed. 1 does not translate the existing line ending. If you do not specify this argument, the system will use the default value of 0.

Calling the Close() method

To call the Close() method:

Type the following JavaScript code:

```
fileobjectname.Close()
```

`fileobjectname` is the name of the file object you created.

The Close method closes the file and releases any locks on the file so that other software can access it.

Calling the Read() method

To call the Read() method:

Type the following JavaScript code:

```
var result = fileobjectname.Read()
```

The Read() method returns the entire file contents as a string into the variable called result. Although line-by-line reading is not available, you can use built-in JavaScript string methods to parse the file.

If another process is concurrently adding information to the file, later calls to the Read method will read the newly added information.

Calling the Write() method

To call the Write() method:

Type the following JavaScript code:

```
fileobjectname.Write("writeoutput" + "\n")
```

`fileobjectname` is the name of the file object you created.

`writeoutput` is the string you want to add to the file.

`\n` adds a new line at the end of the string.

Calling the IsOpen() method

To call the IsOpen() method:

Type the following JavaScript code:

```
var open=fileobjectname.IsOpen()
```

The `var` statement checks to see if the file opening call was successful.

Calling the Exists() method

To call the Exists() method:

Type the following JavaScript code:

```
var exists=fileobjectname.Exists("filepath")
```

`filepath` is the location of the file you are checking.

The `var` statement checks to see if the file exists in the specified folder and returns true if the file is present.

Calling the Delete() method

To call the Delete() method:

Type the following JavaScript code:

```
fileobjectname.Delete("filepath")
```

`filepath` is the location of the file you are deleting. For example, you can type `c:\\fileobjectname.txt`.

Related information

For information about...	See...
JavaScript language	Mozilla Developer Center
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
Using script variables directly in task parameters	"Using simple variables" on page 93
JavaScript task description	"JavaScript" on page 653
Startup and cleanup protocols	"Setting up Startup and Cleanup Protocol processes" on page 75
Setting protocol options	"Setting protocol options" on page 42

Creating new protocol tasks using the JavaScript Wrapper

About the JavaScript Wrapper task

The JavaScript (JS) Wrapper task enables you to create new protocol tasks that have custom task icons and task parameters.

The JS Wrapper task uses the following 3 file types:

- **JavaScript task file (.js)**. The file name must be the same as the JavaScript function and must have the .js file extension. For example, a function named myjavascriptfunction would have the file name, myjavascriptfunction.js.

  This file should contain two parts:

  - The XML command information, which describes the task function, task parameters, and graphical user interface (GUI).
  - The JavaScript function which VWorks will execute.
- **Icon file (.bmp or .jpg)**. The task icon image should be 32x32 pixels. The icon file should have the same file name as the JavaScript function. For example, a function named myjavascriptfunction might have the file name, myjavascriptfunction.bmp.

  If the icon file is missing or not present in OLSS storage, the software uses a default JavaScript icon file.

  Note: The VWorks software already includes some JS Wrapper tasks that use the default JavaScript icon file, such as the JS importCsvTo Inventory task (

).
- **Global script file (jsaddin_global_vars.js)**. This global script file, which is provided with the software, enables you to declare functions and variables that are accessible to a JS Wrapper task or any protocol that includes a JS Wrapper task.

IMPORTANT To minimize any conflict with other global functions and variables, ensure that you use long and unique names for the functions and variables. For example, a naming structure, such as “jsAddin_functionname()” and “jsAddin_functionname_variablename” should result in unique names.

IMPORTANT Ensure that the JavaScript task file name is the same as the JavaScript function, otherwise the task will fail when it executes.

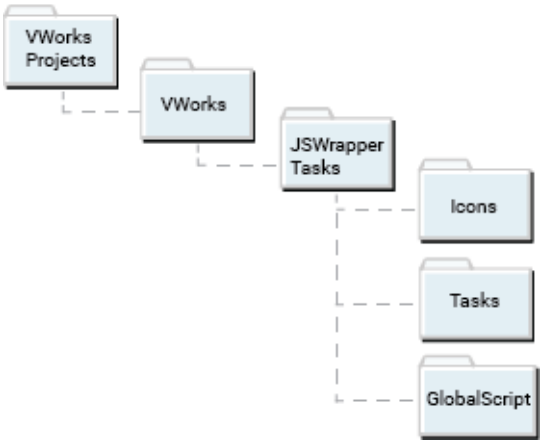
File location requirements

The JavaScript task file and icon file must be placed in specific locations relative to the VWorks Projects folder. The following figure shows the file path for the required files relative to the VWorks Projects folder:

/VWorks Projects/VWorks/JSWrapperTasks/Icons

/VWorks Projects/VWorks/JSWrapperTasks/Tasks

/VWorks Projects/VWorks/JSWrapperTasks/GlobalScript



Ensure that the JavaScript task (.js) file name and the task icon (.bmp or .jpg) are the same as the JavaScript function to be wrapped. For example, for a function named `myjavascriptfunction`, you would have the following task and icon files:

```
/VWorks Projects/VWorks/JSWrapperTasks/Tasks/
myjavascriptfunction.js
```

```
/VWorks Projects/VWorks/JSWrapperTasks/Icons/
myjavascriptfunction.bmp
```

Follow the instructions in "Importing and exporting miscellaneous files" on page 49 to import the JavaScript task file and icon file into Shared Services storage.

VWorks Plus. The files for the JS Wrapper tasks are records of interest. The software logs audit trails for the JavaScript files that you import into the Secure Storage Repository and for the global script file.

The VWorks software uses these files to create the JS Wrapper tasks during startup. Therefore, any changes to the XML command block within the JavaScript file must take place before the VWorks software is started.

JavaScript file

The JavaScript file should have the file extension .js. The file must contain two parts:

1. The XML command information, which describes the task function, task parameters, and graphical user interface (GUI).
2. The JavaScript function which will execute.

JavaScript task file example

Example JavaScript task file to set labware barcode

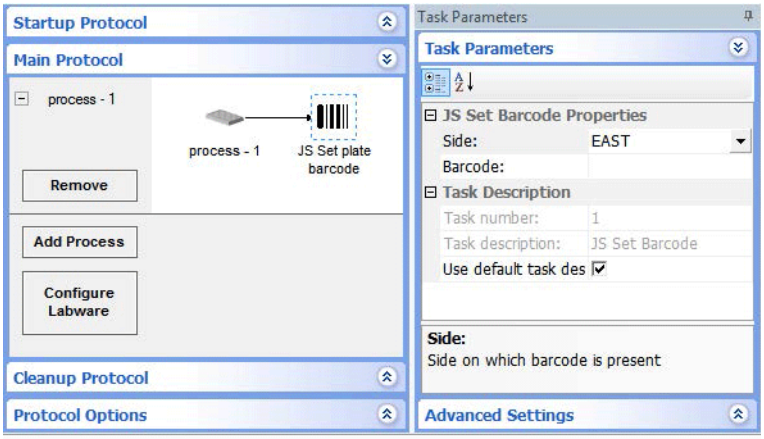
```
///<?xml version='1.0' encoding='ASCII' ?>
///<Velocity11 file='MetaData' md5sum='00000000000000000000000000000000'
version='1.0' >
///     <Command Compiler='0' Editor='2' Name='JS Set Barcode'
Description='JS Set plate barcode' >
///          <Parameters >
///               <Parameter Name='Side' Description='Side on which
barcode is present' Scriptable='0' Type='2' >
///                    <Ranges >
///                         <Range Value='NORTH' />
///                         <Range Value='SOUTH' />
///                         <Range Value='EAST' />
///                         <Range Value='WEST' />
///                    </Ranges>
///               </Parameter>
///               <Parameter Name='Barcode' Description='Value of barcode
to be set' Scriptable='1' Type='1' />
///          </Parameters>
///     </Command>
///</Velocity11>


function jsAddin_setBarcode(jsAddin_setBarcode_parameter1,
jsAddin_setBarcode_parameter2)
{
// convert jsAddin_setBarcode_parameter1 text to variable name
jsAddin_setBarcode_side = eval(jsAddin_setBarcode_parameter1)


// set barcode
plate.setBarcode(jsAddin_setBarcode_side, jsAddin_setBarcode_parameter2)
}
```

Example JavaScript task file to set labware barcode

The preceding code produces a task shown in the following figure.



In this example, the function `jsAddin_setBarcode()` requires two arguments `(jsAddin_setBarcode_parameter1, jsAddin_setBarcode_parameter2)`.

IMPORTANT To minimize any conflict with other global functions and variables, ensure that you use long and unique names for the functions and variables.

The XML block in the file describes the task to be created and how arguments are entered. The following example code shows the basic structure.

Example XML block structure

```
///<?xml version='1.0' encoding='ASCII' ?> XML declaration
///<Velocity11>                           Velocity 11 root element
///      <Command>                        Command element describing the
task
///            <Parameters >              Parameters element describing
how arguments are displayed and entered
///                  <Parameter />        Escaped Parameter element
describing how 1st argument is entered in task parameters GUI
///                  <Parameter />        Escaped Parameter element
describing how 2nd argument is entered in task parameters GUI
///            </Parameters>              Close Parameters element
///      </Command>                       Close Command element
///</Velocity11>                          Close Velocity11 element
```

The XML block uses /// at the start of each line to distinguish it from the JavaScript.

You can find a detailed description of the XML elements in the *VWorks Plugin Developers Guide* in the section on common elements and attributes. The following table provides a summary.

Table Summary of VWorks common XML elements and attributes

Element	Attribute	Value	Description
Velocity11	File	MetaData	Specifies XML structure (in this case, use MetaData)
	md5sum	32-digit hex number	128- bit hash value that can be used to verify the integrity of an XML block. Use 32 zeros until md5sum calculated.
	version	1.0	Currently always 1.0.
Command	Name	String	Name of task (as seen in VWorks Available Tasks list).
	Description	String	Description of task (as seen under the task in a protocol).
	Compiler	Bitmask	Compiler options. 0 indicates no compiler action.
	Editor	Bitmask	Protocol editor options: 0 = disregard (default) 1 = hide if not available 2 = Main protocol editor 4 = Sub-process editor 8 = Startup/cleanup process editor 16 = All editors
Parameters			Contains one or more Parameter elements.

Element	Attribute	Value	Description
Parameter	Name	String	Name of field (parameter).
	Description	String	Description of field (shown in task parameters when field is selected).
	Category	String	A name used to group two or more Parameter elements.
	Hide_if	Logical test	When the value of the conditional expression in the Hide_if attribute is true, the parameter field is hidden or made read-only. Example: `Hide_if='Variable(Type)!=Const(Table)'` The parameter will be hidden if the variable Type (the name of a parameter in the same category) is not equal to the constant (in this case the word Table).
	Scriptable	Integer (boolean)	Indicates if the parameter field can accept JavaScript in the editing dialog. 0 = not scriptable
	Style	Integer	How parameter fields are displayed in the task window. 0 = displayed as read/write (default) 1 = displayed as read only 2 = displayed as read only and hidden when hide disabled tasks selected in VWorks options.

Element	Attribute	Value	Description
Parameter	Type	Integer	Describes the type of field displayed 0 = Provides a Boolean check box. 1 = Allows the user to specify a character string. 2 = Provides a drop- down list box. 3 = Provides a drop- down combo box. 4 = Allows the user to specify a device location. 5 = Allows the user to specify a labware or a fixed location. 6 = Allows the user to specify both a location and the labware to use. 7 = Opens the Well Selection dialog box. 8 = Allows the user to specify an integer. 9 = Allows the user to specify a file path. 10 = Provides a labware drop- down list box. 11 = Provides a liquid- class drop- down list box. 12 = Allows the user to specify a decimal fraction. 13 = Allows the user to specify a file path, where the value can be empty. 14 = Allows the user to enter a password and displays a series of asterisks to hide the password string. 15 = Allows the user to specify an IP address. 16 = Allows the user to select a directory. 17 = Allows the user to enter a time in the format hh:mm:ss. 18 = Refers to an object in the JavaScript scripting context. 19 = Allows the user to enter a date. The format depends on the region and language settings. 20 = Allows the user to enter character strings that can wrap onto multiple lines. 21 = Opens the Pipette Technique Editor. 22 = Opens the Head Mode Selector dialog box. 23 = Describes the tip positions of a tip box. 24 = Opens the Field Composer dialog box. 25 = Displays the available hit pick format files. For example, when the user clicks the down arrow in the Format file field of the Hit pick replicate task, the list that is displayed is of this type.

Element	Attribute	Value	Description
Parameter	Type	Integer	26 = Deprecated. Used to show the available analog input names in the device file where the plugin resides. 27 = Deprecated. Used to show the available digital input names in the device file where the plugin resides. 28 = Deprecated. Used to show the available digital output names in the device file where the plugin resides. 29 = Converts a parameter of this type to, and accesses it as, a JavaScript array object. 30 = Allows the user to specify a duration in the format n Days hh:mm:ss. 31 = Displays a multi- line text box. 32 = Opens the color palette that enables the user to change the colors of various dialog box components.
Ranges			Contains one or more Range elements.
Range	Value	String	When the Parameter Type attribute is 2 or 3 then each Range element and Value attribute describe a menu item that can be selected. When the Parameter Type attribute is 8 or 12 then two Range elements with Value attributes containing integers describe the maximum and minimum possible values.

Designing the task parameters GUI

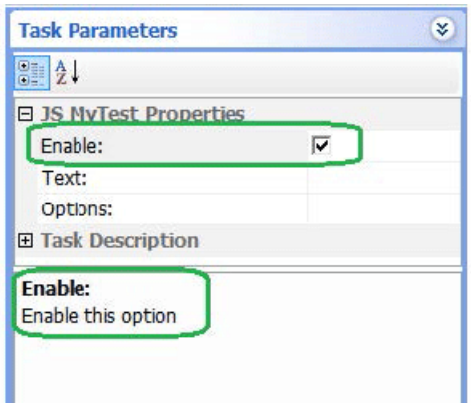
Using different field types

Below are examples of the three most common types of fields for user input.

- **Check box**. When ParameterType ='0', the check box is displayed. When the check box is selected it has a value of 1, and when not selected it has a value of 0.

  For example, the following XML parameter element produces the fields displayed in the following figure.

```
/// <Parameter Name='Enable' Description='Enable this
option' Type='0' Value='1'/>
```

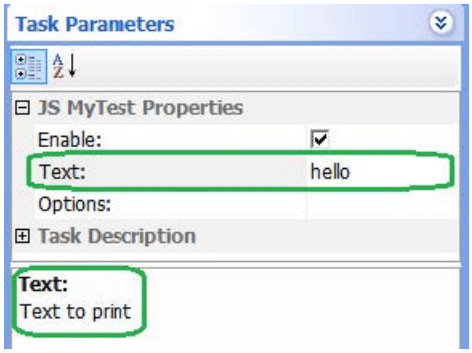


Value attribute specifies the default starting condition. In the example, it is checked.

- **Edit field**. The edit field is displayed for a parameter when Type ='1'. It passes a character string to the functions argument.

  For example, the following XML parameter element displays the fields shown in the following figure.

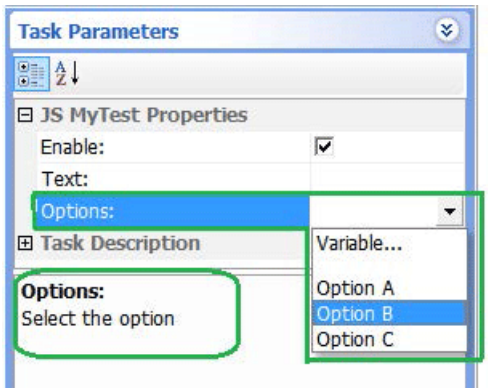
```
/// <Parameter Name='Text' Description='Text to print'
Type='1' />
```



- **Drop list**. The drop list field is displayed for a parameter when Type ='2'. It passes a character string of the selected item to the functions argument. The items in the drop list are defined by the Ranges and Range elements.

  For example, the following XML parameter element produces the display shown in the following figure.

```
/// <Parameter Name='Options' Description=Select the
option' Type='2' >
///          <Ranges>
///                 <Range Value='Option A' />
///                 <Range Value='Option B' />
///                 <Range Value='Option C' />
///          </Ranges>
/// </Parameter>
```



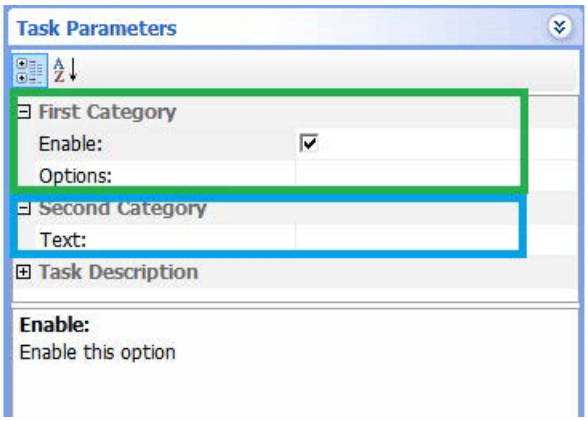
This figure shows the options available.

Grouping fields in the task parameter GUI

You can use the Category attribute to group fields in the task parameters GUI under different headings. The Category attribute places the fields in the order the Categories are first defined and the order each parameter is defined. If no Category is defined for a parameter it is placed in a Category with the name of the task.

For example, the following code produces the display shown in the following figure.

```
/// <Parameter Name='Enable' Description='Enable this
option' Type='0' Value='0' Category='First Category' />
/// <Parameter Name='Text' Description='Text to print'
Type='1' Category='Second Category' />
/// <Parameter Name='Options' Description='Select the
option' Type='2' Category='First Category' >
///          <Ranges>
///                 <Range Value='Option A' />
///                 <Range Value='Option B' />
///                 <Range Value='Option C' />
///          </Ranges>
/// </Parameter>
```



Enabling and disabling GUI fields based on user selections

The Hide_if attribute allows you to enable or disable a field based on the value of another field. The syntax is as follows:

```
Hide_if='[Item1] [logical test] [Item2]',
```

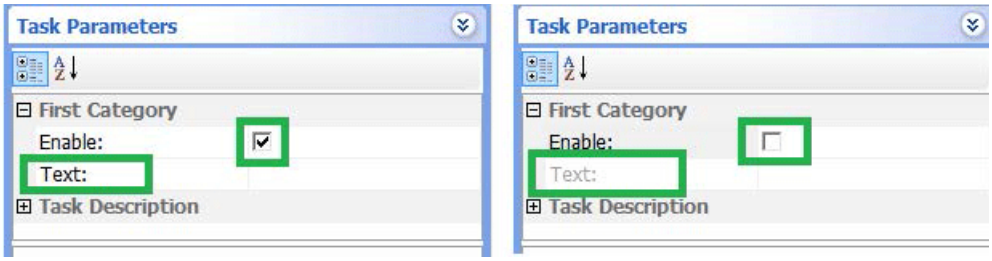
where the field is disabled if the outcome of the logical test is true.

In the following example, the field is disabled if the parameter named Input (this is a variable) is not equal (!=) to the character string "Option A " (this is a constant). The logical tests are similar to JavaScript (==,!=,>,<...).

```
Hide_if='Variable(Input)!=Const(Option A)'
```

The Hide_if attribute only allows variables to be used from within the same Category. If the Enable check box is not selected the Text field is disabled. See the following example.

```
/// <Parameter Name='Enable' Description='Enable this
option' Type='0' Value='0' Category='First Category' />
/// <Parameter Name='Text' Description='Text to print'
Type='1' Category='First Category'
Hide_if='Variable(Enable)==Const(0)'/>
```



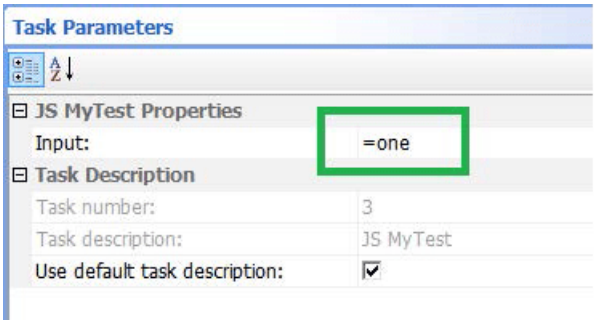
Passing JavaScript variables and arrays to and from the function

You can pass data from JavaScript variables/arrays within the protocol to the JavaScript function of a JS Wrapper task. You can also return data from the JS Wrapper task.

Fields are scriptable

By default fields are scriptable. The field accepts either the ' = javascript variable' or displays the option 'Variable'.

```
/// <Parameter Name='Input' Type='1' />
```



In the this example, the contents of the protocol variable *one* is passed to the JS Wrapper parameter.

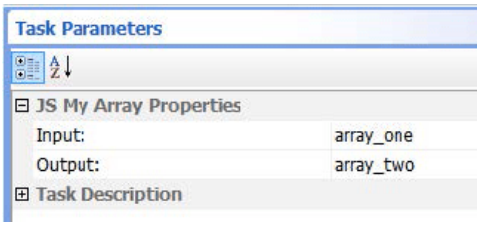
Using the global context

Protocol variables are accessible from within the JS Wrapper function, which allows you to act on them directly. To make JS Wrapper tasks generic (where the names of the protocol variables are not known when writing the JS Wrapper task), you can pass the protocol variable or array names to the JS Wrapper in an edit field during protocol writing, as the following example shows.

Example XML block structure

```
///<?xml version='1.0' encoding='ASCII' ?>
///<Velocity11 file='MetaData' md5sum='00000000000000000000000000000000'
version='1.0' >
///                <Command Compiler='0' Name='JS My Array' Description='JS My
Array' Editor='-1' NextTaskToExecute='1' RequiresRefresh='0'
TaskRequiresLocation='0' VisibleAvailability='1' >
///                 <Parameters>
///                       <Parameter Name='Input' Description='Name of array
to send to function' Type='1' />
///                       <Parameter Name='Output' Description='Name of array
returned from function' Type='1' />
///                 </Parameters>
///                </Command>
///</Velocity11>
function myarray(jsAddin_myarray_input,jsAddin_myarray_output)

{
// Assign input data to temp
eval("jsAddin_myarray_temp =
"+jsAddin_myarray_input)
// Transfer temp to output
eval(jsAddin_myarray_output+" =
jsAddin_myarray_temp")
}
```



Note: array_one and array_two are the names of the variables you want to interact with.

The character string entered for the variable or array name is converted into an actual variable or array name in JavaScript using the function eval().

The function eval() processes a string as JavaScript.

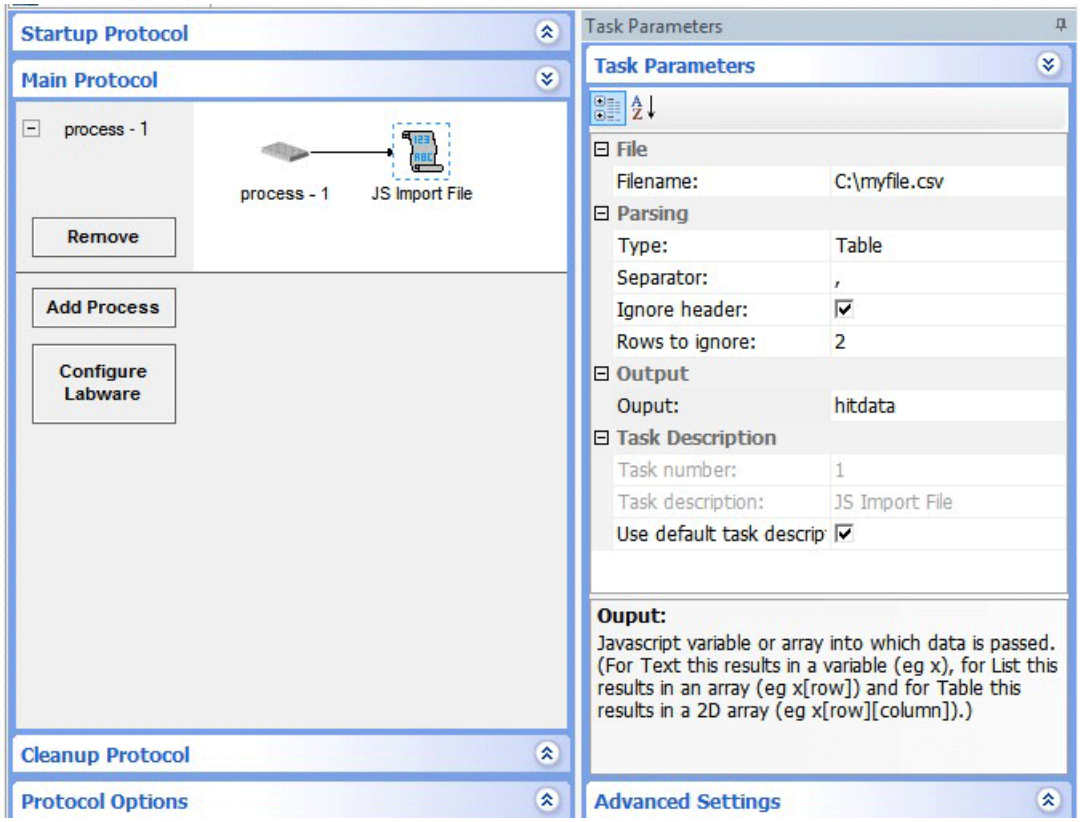
This mechanism allows data to be passed to and from the JS Wrapper task without reserving specific variable or array names.

Example: File import and parsing task

The following example creates a file import and parsing task that has the following key features.

- The import file can be selected.
- The file type can be specified (text, list, or table).
- For the file type table the column separator can be specified (,/;/:/TAB).
- For the file types list and table the header lines can be ignored.
- The output can be specified (name of variable or array).

With an appropriate icon file, the task parameters appears as follows in the user interface.



The following example shows the JS Wrapper file to generate this task.

Example file to generate JS Wrapper task

```
///<?xml version='1.0' encoding='ASCII' ?>
///<Velocity11 file='MetaData' md5sum='00000000000000000000000000000000'
version='1.0' >
///          <Command Compiler='0' Description='JS Import File' Editor='-
1' Name='JS Import File' NextTaskToExecute='1' RequiresRefresh='0'
TaskRequiresLocation='0' VisibleAvailability='1' >
///           <Parameters >
///                  <Parameter Name='Filename' Description='Filename'
Category='File' Scriptable='1' Type='9' />
///                  <Parameter Name='Type' Description='Type of file and
parsing that will be applied. (Text for a single piece of data, List for
data in rows and Table for data in a Grid format (eg CSV).)'
Category='Parsing' Type='2' Scriptable='0' Value='Text'>
///                         <Ranges>
///                                <Range Value='Text' />
///                                <Range Value='List' />
///                                <Range Value='Table' />
///                         </Ranges>
///           </Parameter>
///           <Parameter Name='Separator' Description='Separator for Grid
file type ' Category='Parsing' Type='2' Scriptable='0' Value=','
Hide_if='Variable(Type)!=Const(Table)' >
///                         <Ranges>
///                                <Range Value=',' />
///                                <Range Value='TAB' />
///                                <Range Value=';' />
///                                <Range Value=':' />
///                         </Ranges>
///           </Parameter>
///          <Parameter Name='Ignore header' Description='Ignore header
rows in list and grid files.' Category='Parsing' Type='0' Scriptable='1'
Hide_if='Variable(Type)==Const(Text)' />
///          <Parameter Name='Rows to ignore' Description='Number of rows
to ignore in list and grid files.' Category='Parsing' Type='1'
Scriptable='1' Hide_if='Variable(Type)==Const(Text)' />
///          <Parameter Name='Ouput' Description='Javascript variable or
array into which data is passed. (For Text this results in a variable
(eg x), for List this results in an array (eg x[row]) and for Table this
results in a 2D array (eg x[row][column]).)' Category='Output' Type='1' />
///           </Parameters>
///          </Command>
///</Velocity11>
```

Example file to generate JS Wrapper task

```
Function
jsAddin_ImportFile(jsAddin_ImportFile_filename,jsAddin_ImportFile_filetype
,jsAddin_ImportFile_separator,
jsAddin_ImportFile_ignoreheaders,jsAddin_ImportFile_rowstoignore,jsAddin_I
mportFile_output)
{
// create file object jsAddin_ImportFile_fo = new File()
// open file jsAddin_ImportFile_fo.Open(jsAddin_ImportFile_filename)
// read file into variable contents jsAddin_ImportFile_contents =
jsAddin_ImportFile_fo.Read()
// close file jsAddin_ImportFile_fo.Close()
// Parse contents depending file type/parsing method
switch(jsAddin_ImportFile_filetype)
{
// Text - assumes one value and passes directly to variable
jsAddin_ImportFile_temp case "Text":
jsAddin_ImportFile_temp = jsAddin_ImportFile_contents
break;
// List - assumes a list with a single piece of data on each row case
"List":
jsAddin_ImportFile_temp = jsAddin_ImportFile_contents.split("\n")
break;
// Table - assumes a tabular data structure in row and columns case
"Table":
jsAddin_ImportFile_temp = new Array()
jsAddin_ImportFile_lines = jsAddin_ImportFile_contents.split("\n")
if(jsAddin_ImportFile_ignoreheaders=="1")
{
jsAddin_ImportFile_start = jsAddin_ImportFile_rowstoignore
}
else
{
jsAddin_ImportFile_start = 0
}
jsAddin_ImportFile_ii = 0
for(jsAddin_ImportFile_i=jsAddin_ImportFile_start;jsAddin_ImportFile_i<jsA
ddin_ImportFile_lines.length;jsAddin_ImportFile_i++)
{
jsAddin_ImportFile_temp[jsAddin_ImportFile_ii]=jsAddin_ImportFile_lines[js
Addin_ImportFile_i].split(jsAddin_ImportFile_separator)
jsAddin_ImportFile_ii++
}
break;
}
// assign jsAddin_ImportFile_temp to output variable
eval(jsAddin_ImportFile_output +" = jsAddin_ImportFile_temp")
}
```

Global script file

The global script file (jsaddin_global_vars.js) allows you to declare functions and variables that will be accessible to a JS Wrapper task or any protocol that has a JS Wrapper task.

Note: The software stores the global script file in the following folder: /VWorks Projects/VWorks/JSWrapperTasks/GlobalScript

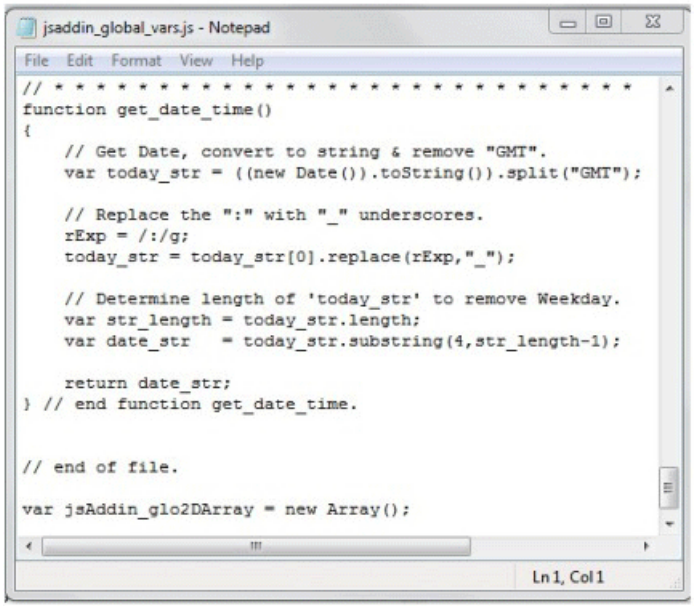
Before launching the VWorks software, you can add functions and variables to the global script file, including the open() command to process other JavaScript files.The functions and variables defined in the global script file are available to any protocol that contains a JS Wrapper task. Unlike the functions and variables described in a JS Wrapper task, the ones that are defined in the global script file are available before the JS Wrapper task is executed.

Example of using global script file

In this example, both the JavaScript tasks contain the same code, as shown in the Task Parameters area.



Global script file



The function `get_date_time()`, which is part of the global script file is available both before and after the execution of the JS Wrapper task (JS Set plate barcode).

The function `jsAddin_setBarcode()`, which is part of the JS Wrapper task (JS Set plate barcode), is only available after the JS Wrapper task is executed.

Example of using global script file

Sequence of events displayed in Main log

		Main protocol starting
process - 1 1		Process starting, north barcode: No bar code, east barcode: No bar code, south barcode: No bar code, west barcode: No bar code
		Dec 15 2012 13_03_31
		script:1:ReferenceError: jsAddin_setBarcode is not defined
process - 1 1		After 1 second...
process - 1 1	1	JavaScript
process - 1 1		After 6 seconds...
process - 1 1	1	Completed: JavaScript
process - 1 1		After 6 seconds...
process - 1 1	2	JS Set plate barcode
process - 1 1		After 11 seconds...
process - 1 1	2	Completed: JS Set plate barcode
		Dec 15 2012 13_03_32
		1234
process - 1 1		After 11 seconds...
process - 1 1	3	JavaScript
process - 1 1		After 16 seconds...
process - 1 1	3	Completed: JavaScript
process - 1 1		Process finishing

IMPORTANT To minimize any conflict with other global functions and variables, ensure that you use long and unique names for the functions and variables. For example, a naming structure, such as “jsAddin_functionname()” and “jsAddin_functionname_variablename” should result in unique names.

Enabling global scripts

The Global script file is only processed when the protocol contains a JavaScript Wrapper task.

If you want to use the Global script file and not a specific JavaScript Wrapper task, you can prepare a JavaScript Wrapper task whose sole function is to enable the Global script file. Create a JavaScript wrapper file called jsAddin_enableGlobalScripts.js containing the following.

Example of JS Wrapper file to enable Global script file

```
///<?xml version='1.0' encoding='ASCII' ?>
///<Velocity11 file='MetaData' md5sum='00000000000000000000000000000000'
version='1.0' >
///      <Command Compiler='0' Editor='16' Name='JS Enable Global Scripts'
Description='JS Enable Global Scripts File (jsaddin_global_vars.js)' >
///            <Parameters />
///      </Command>
///</Velocity11>
function jsAddin_enableGlobalScripts()
{
print("Global Script File Enabled")
}
```

Related information

For information about...	See...
Using JavaScript in the VWorks software	"Using JavaScript" on page 100
Using script variables directly in task parameters	"Using simple variables" on page 93
JavaScript task description	"JavaScript" on page 653

4 Creating forms for protocol operators

You can create forms with custom user interfaces for the operators who run your protocols. A form provides a simplified interface, which can be helpful for users who have limited or infrequent experience with the VWorks software.

You must have VWorks administrator or technician access to create protocol forms, and you should also have an understanding of how to create VWorks protocols.

This chapter contains the following topics:

- "About forms for running protocols" on page 178
- "Workflow for creating or editing a form" on page 181
- "About the naming of form controls" on page 184
- "Creating or editing a form" on page 185
- "Overview of the specialized buttons" on page 188
- "Configuring a run button and other specialized buttons" on page 194
- "Adding indicators for elapsed time and progress to a form" on page 202
- "Adding form controls that allow editing or runtime data display" on page 204
- "Example: Creating a scripted Pushbutton control in a form" on page 232
- "Setting the form properties" on page 240
- "Understanding JavaScript context in form design" on page 243
- "Opening a message from a form button using JavaScript" on page 247
- "Configuring a form for experiment IDs and Experiments database" on page 249
- "Adding specialized controls for the Experiments Editor" on page 253
- "Configuring a protocol to track experiment data" on page 260

About forms for running protocols

About this topic

This topic describes VWorks forms and the Form Designer window.

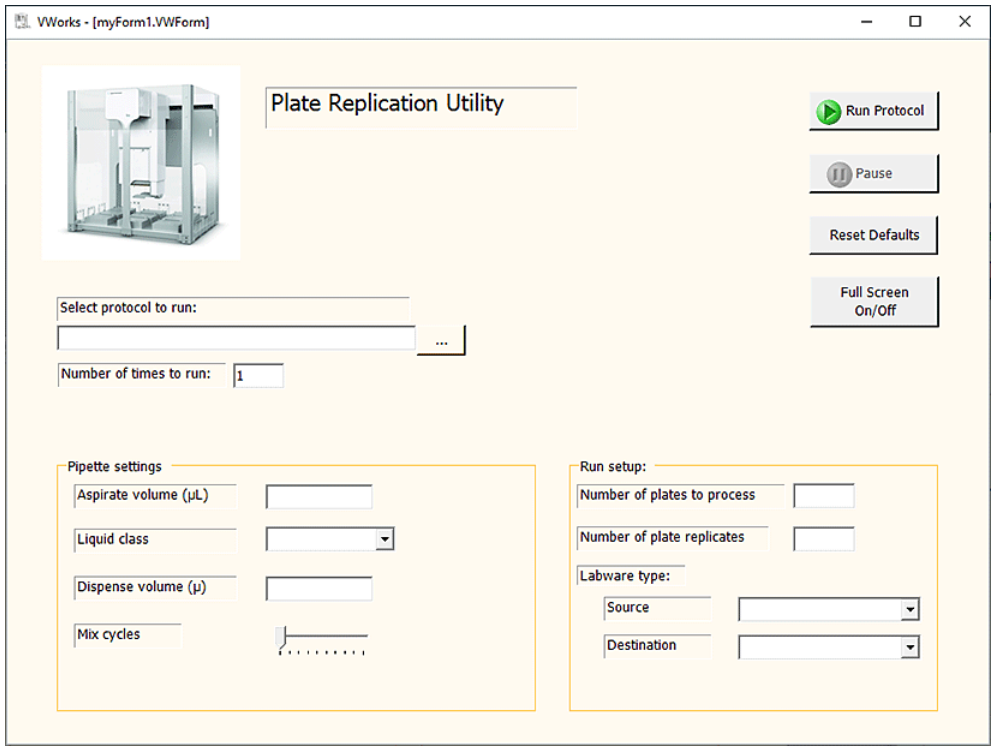
You must be an administrator or technician to create or edit forms. You should also have an understanding of how to create protocols in the VWorks software.

Forms defined

A form is a customized user interface that enables operators to run an associated VWorks protocol. A VWorks user who creates protocols can design forms to simplify the operator duties when it comes time to run the associated protocols.

The form can be simple or relatively complex, depending on the requirements of a target group of operators. At a minimum, the form should have buttons for starting and pausing the associated protocol. You may also want to include the Full Screen On/Off button (default) to enable the form users to toggle the VWorks window between a full screen form view and the normal tabbed view. The form design can allow the operator to enter task parameter values for the associated protocol or limit the operator to a read-only display of task parameter values, progress, and other protocol data that the form's author wants to display.

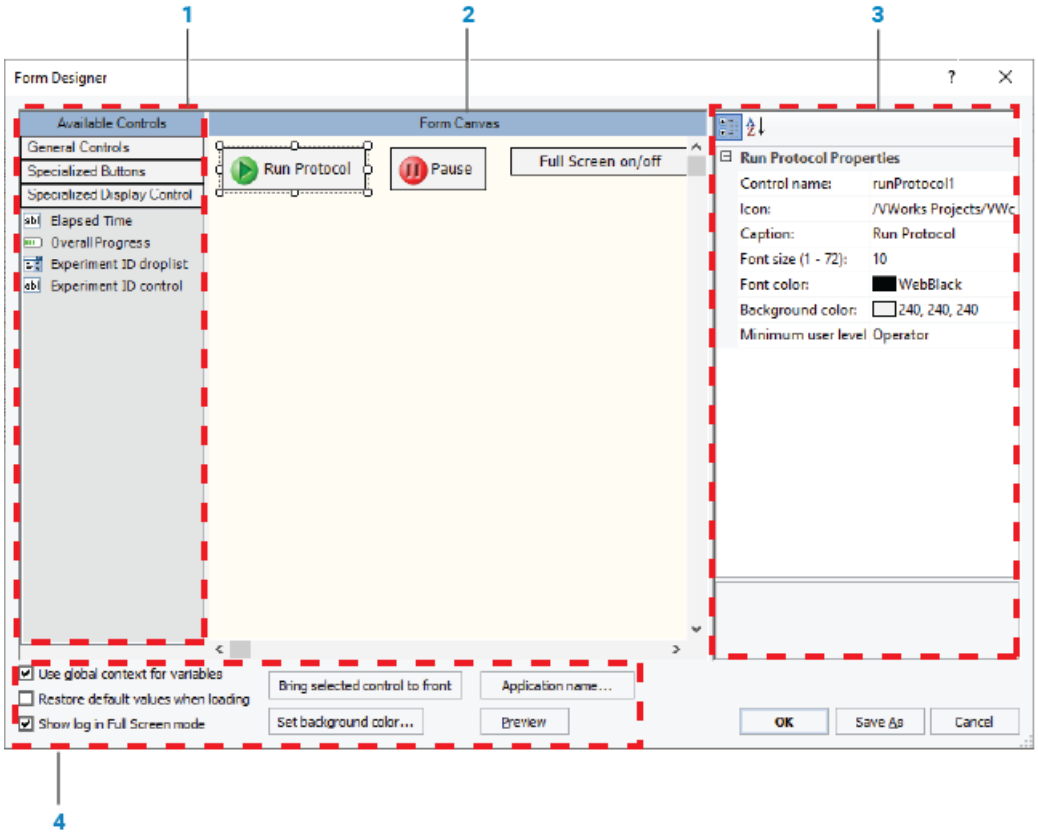
Figure Example of VWorks window displaying a form in full screen view without the logs



Form Designer overview

The following figure shows four primary areas in the Form Designer window, and the table describes these four areas.

Figure Areas in Form Designer window



Item	Area	Description
1	Available Controls	The control palette, which includes three categories from which to select controls and indicators.
2	Form Canvas	The work area where the form's author arranges the form controls while viewing a representation of the form. Three buttons are included in this area by default: Run Protocol, Pause, and Full Screen on/off. As you drag controls into position in the Form Canvas, invisible grid lines assist with the alignment based on the controls that are already in the form. *Note:* To disable the autogrid, press ALT while dragging the controls into place.
3	Control Properties	The properties associated with a control that is currently selected in the Form Canvas. The figure shows the Run Protocol Properties because the Run Protocol button is selected.

Item	Area	Description
4	Form properties	The properties that apply to the entire form, such as the background color and the context for JavaScript variables that are used in the form.

Available controls overview

The Available Controls area of the Form Designer window contains the following areas, each of which provides different types of controls that you can use in your form design:

- **General Controls area**. Most of the controls in the General Controls area can be assigned a JavaScript variable. By assigning a JavaScript variable, you can allow form users to edit task parameter values or enable the display of runtime data. The JavaScript variable must be assigned in both the control properties and in the corresponding task parameter in the protocol.

  For details about the general controls and how to configure them, see "Adding form controls that allow editing or runtime data display" on page 204.
- **Specialized Buttons area**. The Specialized Buttons area contains button controls that are already programmed to perform commands that are currently available elsewhere in the VWorks window, such as Run Protocol and Pause. For details about the buttons and how to configure them, see "Overview of the specialized buttons" on page 188.
- **Specialized Display Controls area**. The controls in the Specialized Display Controls area display data that is not drawn from a JavaScript variable. These controls include Elapsed Time, Overall Progress, and Experiment ID. For details about the specialized display controls and how to configure them, see "Adding indicators for elapsed time and progress to a form" on page 202.

Related information

For information about...	See...
Creating or editing a form	"Workflow for creating or editing a form" on page 181
Controls under General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Controls under Specialized Buttons in the Form Designer window	"Overview of the specialized buttons" on page 188
Controls under Specialized Display Controls in the Form Designer window	"Adding indicators for elapsed time and progress to a form" on page 202
Form properties in the Form Designer window	"Setting the form properties" on page 240
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22

For information about...	See...
Working with JavaScript	"Creating a protocol: advanced topics" on page 89

Workflow for creating or editing a form

About this topic

To create a form, you must be an administrator or technician and have an understanding of how to create protocols in the VWorks software.

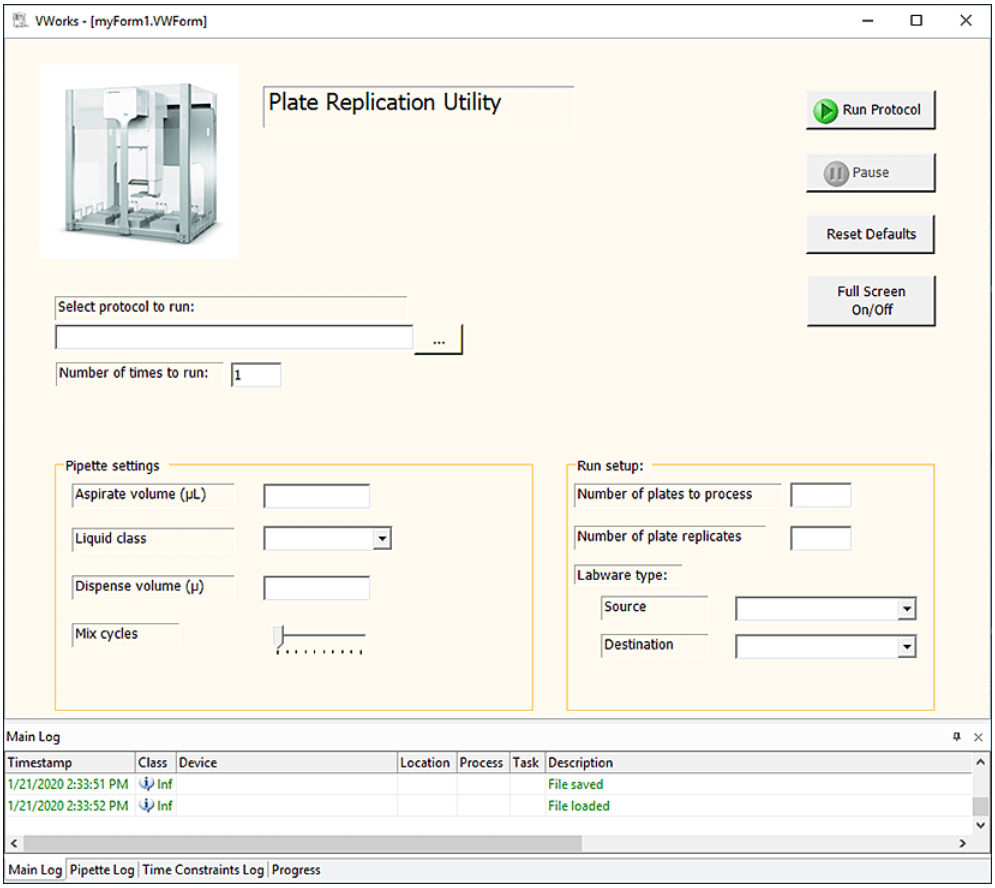
Before you start

Consider the following:

- Create the protocol (.pro) or runset (.rst) that you want the form to run.
- Determine how you want the form to function. For example, will users simply press a run button to start the associated protocol, or will they be required to make selections in the form before the run?
- If the form will include controls that display runtime data for a given task or require a user to enter values, you must have a JavaScript variable associated with the control and the corresponding task parameter. For details on JavaScript variables, see "Using simple variables" on page 93.
- Determine whether the form users will require documentation specific to your form.
- If you want to track the runtime values from the form settings in an experiments database, see "Configuring a form for experiment IDs and Experiments database" on page 249
- *VWorks Plus only*. Consider the record's state. VWorks logs audit trails for forms and protocols based on their record state:
  - To run a protocol successfully using the form's Run Protocol button or when running a protocol that specifies the form to use under Protocol Options, the form must be in a state equal to or greater than the protocol. For example, if the protocol is in validation, the form file must be in a validation or released state.
  - To run a protocol successfully using a Run Specified Protocol button on a form, the specified protocol must be in a state equal to or greater than the form.

  Note: The software logs an audit trail of changes to a form file (.VWForm), not the runtime values specified in the form.

Figure Example VWorks form in full screen view with logs



Workflow

Step	For this task...	See...
1	Determine the naming schema for the form Control name properties so that control references in the Main log and in audit trail reports are meaningful.	"About the naming of form controls" on page 184
2	Import any image or icon files you want to use in the form. You save the imported file to a folder in Shared Services storage, for example, /VWorks Projects/VWorks/Forms/Images	"Importing and exporting miscellaneous files" on page 49
3	Open the Form Designer.	"Creating or editing a form" on page 185
4	Select an appropriate button to run the protocol or runset.	"Overview of the specialized buttons" on page 188

Step	For this task...	See...
5	Determine whether to include the Pause and Full Screen on/off buttons (default), and add any other specialized buttons that you want to use.	"Overview of the specialized buttons" on page 188
6	Add indicators for the elapsed time and progress of the protocol or runset that the form is running.	"Adding indicators for elapsed time and progress to a form" on page 202
7	Add controls that display or allow editing of the run data.	"Adding form controls that allow editing or runtime data display" on page 204
8	Set the form properties and save the form.	"Setting the form properties" on page 240

Related information

For information about...	See...
Creating or editing a form	"Workflow for creating or editing a form" on page 181
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Working with JavaScript	"Creating a protocol: advanced topics" on page 89
JavaScript context and form design	"Understanding JavaScript context in form design" on page 243

About the naming of form controls

IMPORTANT Ensure that you use meaningful nomenclature for the Control name property of any button controls and for controls that allow form users to enter or edit values. The software uses the Control name property when logging changes in the VWorks Main Log and if an audit trail of changes is logged for the form.

Control name versus Caption property

Form controls have the following nomenclature properties:

- *Control name property*. The descriptive name associated with the control. Every control has the Control name property.

  Although the Control name does not appear on the form, it does appear in events logged in the Main Log and audit trail reports.
- *Caption property*. The label for a given control. The Caption appears on the form. A subset of controls have the Caption property, such as the Pushbutton and Checkbox controls and the Specialized Buttons. You can use a Static text control to provide a label for a control that does not have the Caption property.

Nomenclature guidelines for Control name property

By default, the software uses a generic name and a sequential number to assign the Control name property. So, multiple instances of a given control use the following naming convention:

genericName1, genericName2, genericName 3, and so forth

The following table lists the default value and some descriptive alternatives for multiple instances of controls on a single form.

Form control	Control name property	
	Default value	Descriptive examples
Edit control	operatorConsoleTextBox1 operatorConsoleTextBox2 operatorConsoleTextBox3 operatorConsoleTextBox4	AspirateVolume DispenseVolume MixCycles NumberOfRuns
Pushbutton	operatorConsoleButton operatorConsoleButton2 operatorConsoleButton3 operatorConsoleButton4	Method_Load_Button Method_Save_Button ClearAll Button LiteraturePage_Button
Run Protocol	runProtocol1 runProtocol2 runProtocol3	RunProtocol_Button RunTipTransfer_Button StartRun_Button

As the examples show, the default nomenclature for the Control name property are too generic for the corresponding events captured in the Main Log or logged in an audit report. To ensure that the logged events are recognizable, you should create a set of nomenclature guidelines for the Control name property to ensure that descriptive and meaningful names are used in your forms.

Related information

For information about...	See...
Audit trails and audit trail reports	*VWorks Automation Control Setup Guide*
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
Controls under General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Controls under Specialized Buttons in the Form Designer window	"Overview of the specialized buttons" on page 188

Creating or editing a form

About this topic

To open the Form Designer, you must be an administrator or technician.

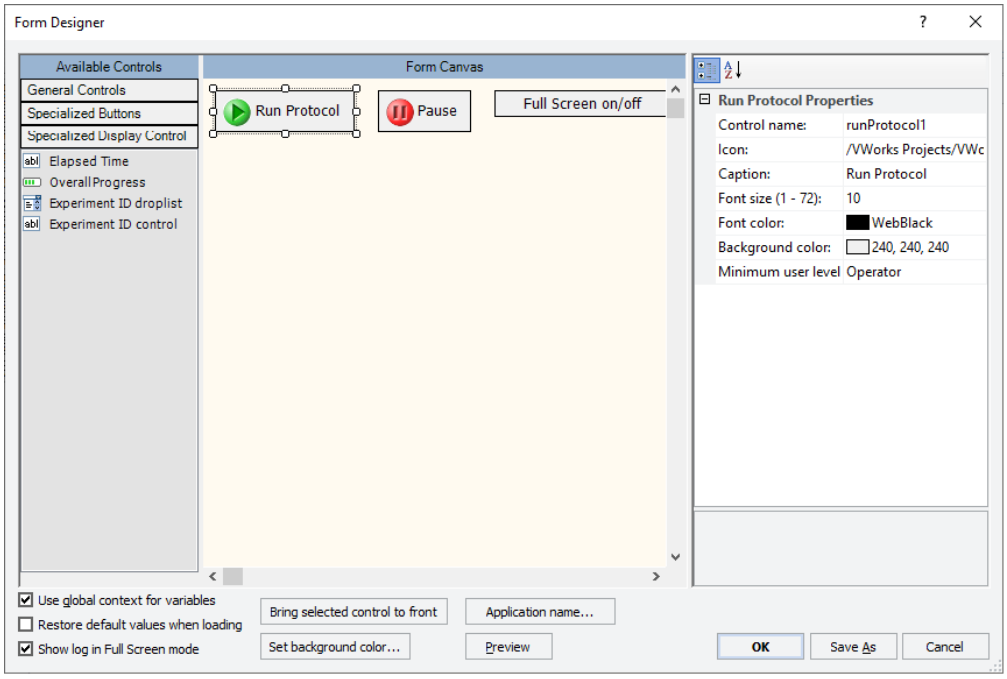
This topic describes how to open the Form Designer, create a form, edit a form, and how to move and resize the window.

Creating a new form

To open the Form Designer and create a new form:

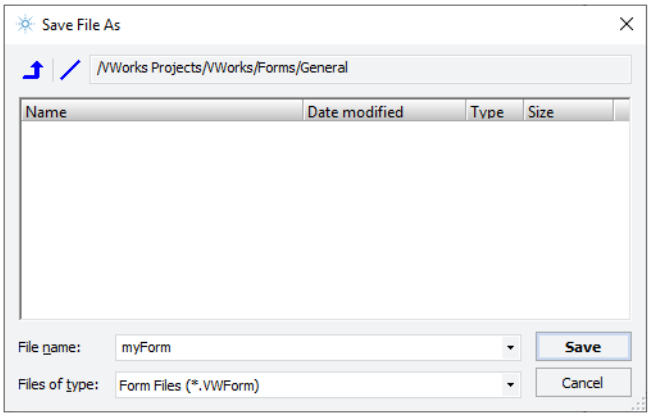
1 In the VWorks window, click **File > New > Form**. The Form Designer window opens, and three buttons appear by default in the Form Canvas area.

Figure Form Designer window with the three default buttons



2 In the **Form Designer** window, click **Save As**.

3 In the **Save File As** dialog box, type the file name, and then click **Save**.



4 *VWorks Plus only*. If an audit trail is being logged, select or type a reason for the change in the **Audit Comment** dialog box, and then click **OK**.

Editing a form

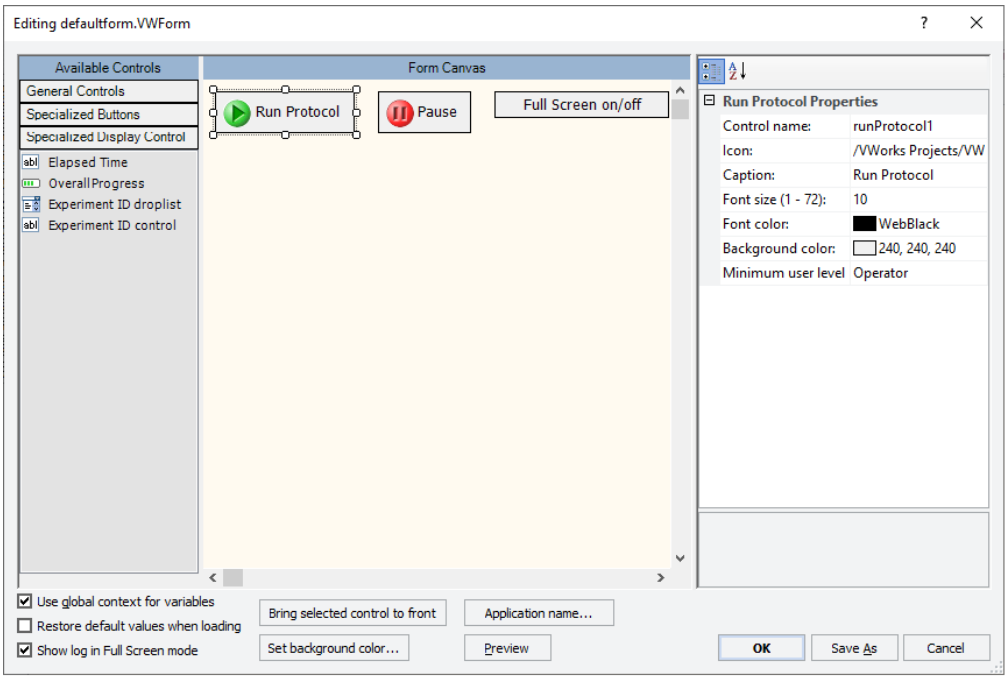
To open the Form Designer and edit a form:

1 In the **VWorks** window, choose **File > Open Form**. In the **Open File** dialog box, select the .VWForm file that you want to edit, and then click **Open**.

The form tab appears in the protocol area of the VWorks window.

2 Choose **Tools > Edit Form**. The selected form appears in the Editing .VWForm window.

Figure Editing .VWForm window with a form to be edited



3 When you finish editing, click **OK** to save the form.

4 *VWorks Plus only*. If an audit trail is being logged, select or type a reason for the change in the **Audit Comment** dialog box, and then click **OK**.

Moving and resizing the window

You can move the Form Designer window to the side of the screen so that you can continue to access the protocol task parameters while designing your form.

To move the Form Designer window:

Drag the Form Designer window to the new location.

To resize the Form Designer window:

Move the pointer to an edge of the window. When the pointer turns into a double-sided arrow, drag the edge of the window to the new size.

To resize the areas inside Form Designer window:

Move the pointer over the vertical splitter bar that separates each area. When the pointer turns into a double-sided arrow, drag the splitter to the new position.

Related information

For information about...	See...
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
Forms and Form Designer overview	"About forms for running protocols" on page 178

For information about...	See...
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Overview of the specialized buttons

About this topic

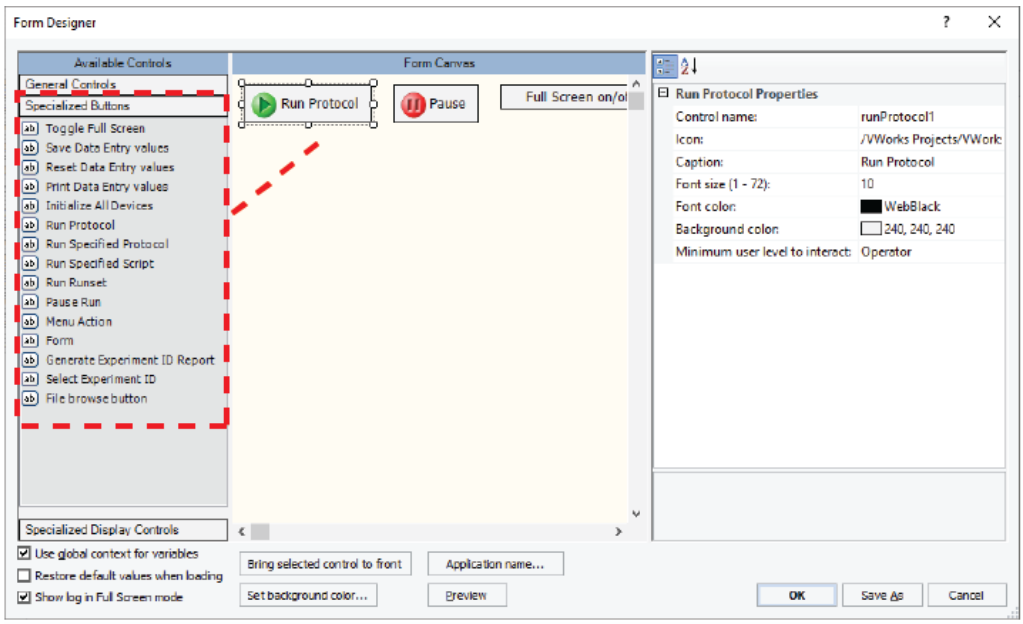
To create or edit forms, you must be an administrator or technician. You should also have an understanding of how to create protocols in the VWorks software.

This topic describes the following:

- "Buttons for data entry values" on page 189
- "Determining which run button to include" on page 189
- "About the Pause Run and Toggle Full Screen buttons" on page 190
- "About the Menu Action button" on page 193
- "About other specialized buttons" on page 193
- "Configuring a run button and other specialized buttons" on page 194

The Form Designer window contains buttons that perform specific commands, which are available elsewhere in the VWorks window. You can find these buttons under Specialized Buttons in the Form Designer window.

Figure Form Designer window Specialized Buttons area and the default buttons



For more details on creating forms, see "Workflow for creating or editing a form" on page 181.

To configure the specialized buttons, see "Configuring a run button and other specialized buttons" on page 194.

Buttons for data entry values

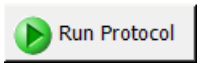
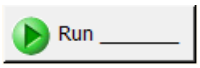
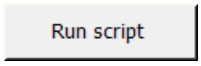
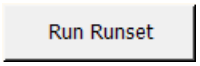
The Data entry property, which enables form users to enter or edit values, is available for many of the Form Designer controls found under General Controls. The following buttons manage the data entry values that are entered by a form user.

Button	Description
Print Data Entry values	Prints all data entry values associated with the controls in the form, including values entered by the form user and default values that were not modified.
Reset Data Entry values	Resets all data entry values associated with the controls in the form to the default values. Alternatively, values can be reset automatically when the form is reopened if the Restore default values when loading check box is selected in the Form Designer window when creating the form.
Save Data Entry values	Saves the data entry values associated with the controls in the form so that the values persist until they are explicitly reset.

Determining which run button to include

The following table describes the run button options that you can use on your form. To configure these run buttons, except the Pushbutton control, see "Configuring a run button and other specialized buttons" on page 194.

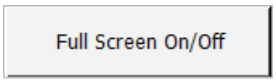
Run buttons	Description
Pushbutton control	Provides the most flexible way to run a protocol using a form, because you can allow the form users to choose which protocol to run and the number of run times. The Pushbutton control is under the General Controls area. For details on how to configure the Pushbutton control, see: • "Adding form controls that allow editing or runtime data display" on page 204 • "Example: Creating a scripted Pushbutton control in a form" on page 232.

Run buttons	Description
Run Protocol (default) 	Opens the Run Configuration Wizard for the protocol, so that the user can enter the number of times to run, when to start the run, and the starting barcode. When the user clicks Finish in the wizard, the protocol starts, or is scheduled to start in the future. The Run Protocol button is available to the form user only if the protocol is open and no run is currently in progress. The form can be used to start different protocols, but only one instance of the form can be open and running a given protocol at a time. *Setup requirements*. In the Protocol Options area of the VWorks window, you must specify the form to use (.VWForm). You can also choose to automatically load the form file, so that the form opens automatically when the protocol is opened. If you export a protocol that has a specified form to use, the form file can be included in the exported files.
Run Specified Protocol 	Starts the run for the specified protocol immediately, unless you include a script for prerun checks. The Run Configuration Wizard does not appear in this case. If the properties include a script for prerun checks, the checks are performed before the protocol runs. The Run Specified Protocol button is grayed the instant it is pressed, and remains grayed until either the checks fail or the protocol completes. *Setup requirements*. The properties for the button must specify the protocol file (.pro) and the number of times to run it.
Run Specified Script 	Runs the specified JavaScript code. For example, the script could download an Excel file (.xlsx) from storage, run the downloaded file, and then upload the file to storage again. For details about this example, see "VWorks-defined functions" on page 103. Alternatively, you can use the Run Specified Protocol button to specify a script for prerun checks and run a specified protocol. See also "Using JavaScript" on page 100.
Run Runset 	Starts the specified runset. The Run Configuration Wizard does not appear in this case. If any runs are scheduled to start as soon as possible, a warning message asks the operator to make sure that the system is ready for the runs to start. *Setup requirements*. The properties for the button must specify the runset file (.rst), and you must select the Use global context for variables check box for the form.

About the Pause Run and Toggle Full Screen buttons

IMPORTANT Be sure to include the Toggle Full Screen button in your form if you want the user to view the form in the full screen view.

When you open the Form Designer to create a form, the Pause Run and Full Screen on/off buttons are included in the form by default. If you accidentally remove these buttons, you can reselect the Pause Run and Toggle Full Screen controls from the Specialized Buttons area of the Form Designer window.

Buttons	Description
Pause Run (default) 	Provides the same function as the Pause all button on the toolbar. The button pauses the run and opens the Scheduler Paused dialog box. For details, see "Pausing the run" on page 337. Include the Pause button in your form to give the form users a way to pause, continue, or abort the run when the form is in full-screen view and the toolbar is hidden.
Toggle Full Screen (default) 	Changes the VWorks window between the normal view (full screen off) and full-screen view of the form: • *Full screen on*. Displays only the form in the VWorks window. Optionally, the view can include the Main Log, Pipette Log, Time Constraints Log, and Progress tabs. All other controls and areas, such as menus, toolbars, and the work area are hidden. • *Full screen off*. Displays the form as a tab in the protocol area of the VWorks window.

IMPORTANT

The full-screen view is an option only if the form includes the Toggle Full Screen button. Also, if the form's user clicks the button in either full screen on or off view, the VWorks software closes.

The following figures show examples of the full screen on and off.

Figure VWorks window with example form in full screen view

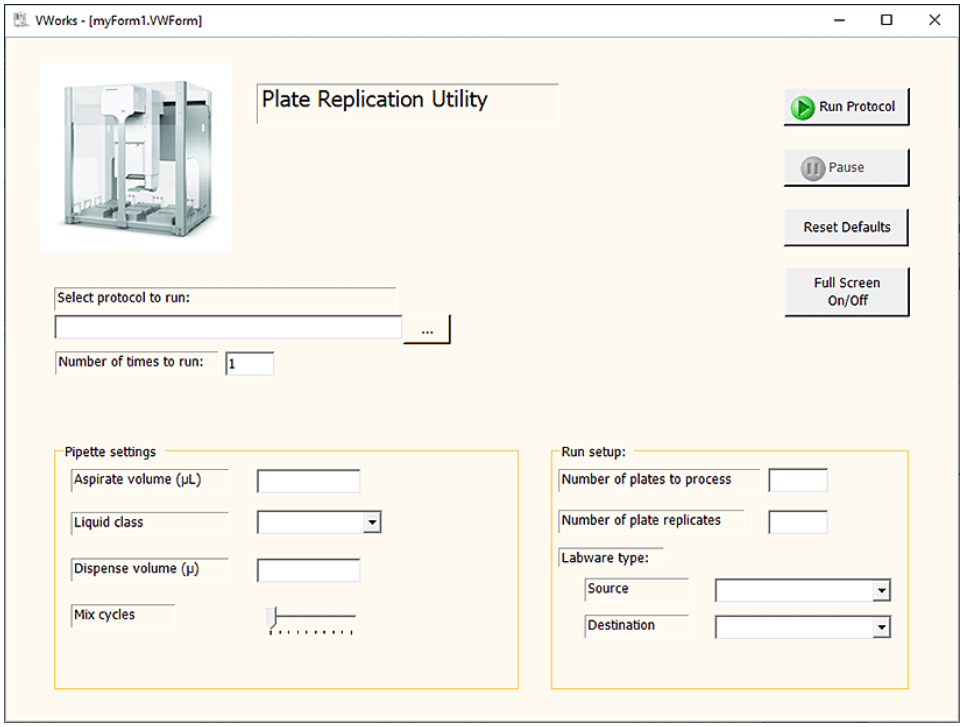
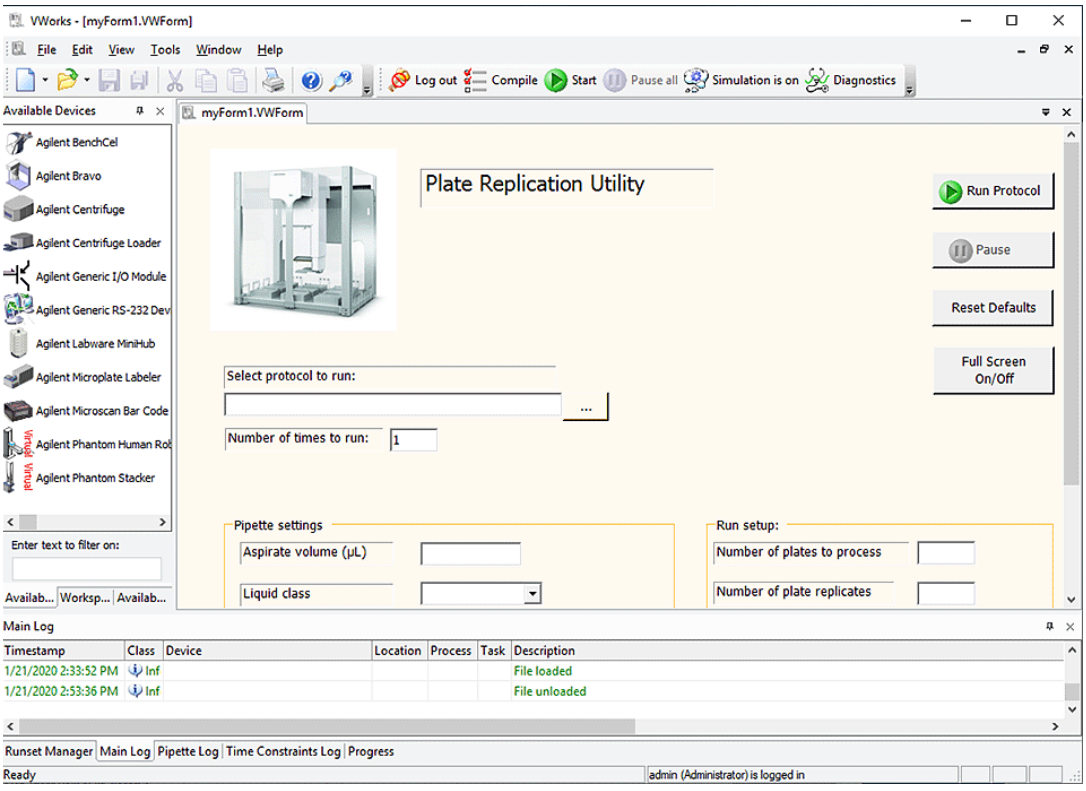


Figure VWorks window with example form in tab view (full screen off)



About the Menu Action button

The Menu Action button performs a command that the form designer selects from a list of commands, most of which are also available in the Tools menu of the VWorks window.

IMPORTANT

The Menu Action button may give a VWorks operator the ability to perform actions that normally require technician-level access. If you want to prevent this, set the **Minimum user level to interact** property to Technician or Administrator.

About other specialized buttons

Button	Description
Initialize All Devices button	Performs the same function as the Initialize all devices button in the Devices tab of the VWorks window. Both buttons open the selected device file (.dev) and establish communication with the corresponding devices. If you want the form user to be able to initialize the devices manually while the form is displayed in full-screen view, include the Initialize All Devices button on the form. *Note:* If the simulation mode is turned off when you open a protocol, the software automatically loads the device file associated with the protocol and asks you whether you want to initialize the devices.
Form	Opens a specified form. This button provides the option to close the current form when the form that the Form button specifies opens.
Generate Experiment ID Report	Opens the Experiments Editor window. For details, see "Configuring a form for experiment IDs and Experiments database" on page 249.
File Browse button	Opens a Open File or Save File As dialog box in a similar manner as the File Browsing control, but without the file name display. The File Browse button can be customized in the same manner as other specialized buttons. For details, see "File browsing control and File Browse button properties" on page 223.

Configuring a run button and other specialized buttons

Procedure

Note: If you want one control to overlap another control on the form, select the control that you want on top, and then click **Bring selected control to front**.

To configure the specialized buttons:

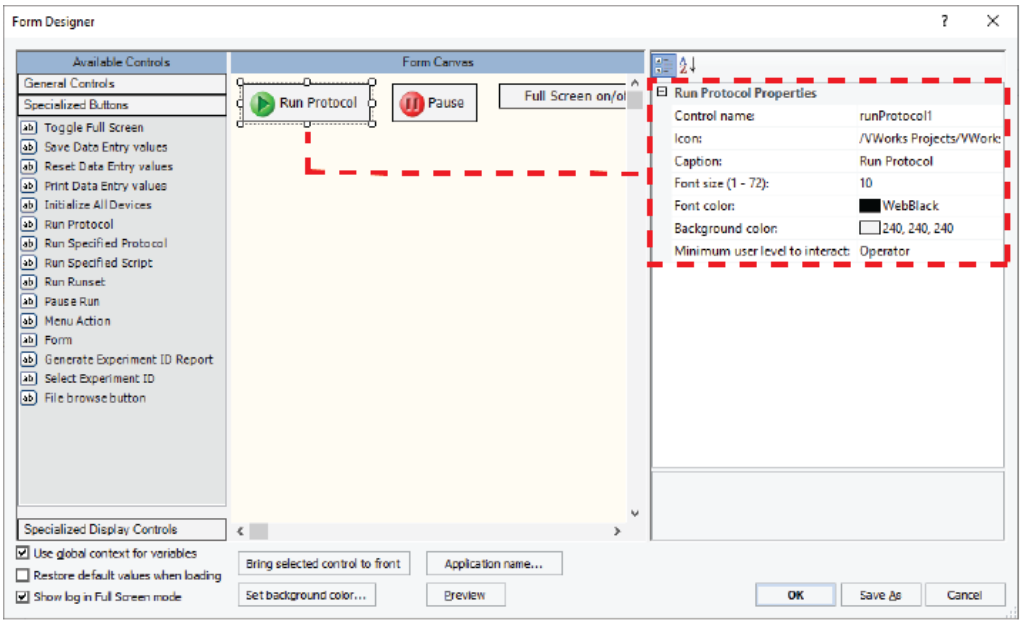
1 If the desired button is not yet in the form, click **Specialized buttons**, and drag the button to the **Form Canvas** area.

Note: To turn off the autogrid, press ALT while dragging the control into position.

2 In the **Form Canvas** area, click the button that you are configuring. A resizing box appears around the button's border. To resize the button, drag one of the sizing handles.

The corresponding properties appear for the selected button as the following figure shows.

Figure Form Designer window displaying properties for the selected button

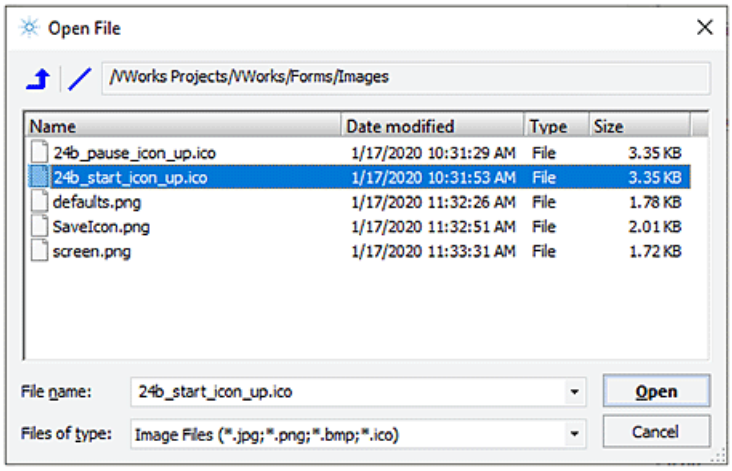


Note: Click **Preview** to display the image that the form user will see.

3 Edit the **Properties** of the button as required.

Note: For the File Browse button, see "File browsing control and File Browse button properties" on page 223.

Properties specialized buttons have in common

Property	Description
Control name	The descriptive name associated with the control. The Control name does not appear on the form. Use a functionally descriptive name so that references to this control are recognizable in the Main Log and in audit trail reports. For details, see "About the naming of form controls" on page 184.
Icon	*Optional*. You may select an image to use as an icon in addition to the button caption. Image files must be imported into the VWorks file storage system before they are available for use in the VWorks software. To use an image in the form: **1** If the image file is not yet in the VWorks file storage, import the image file. See "Importing and exporting miscellaneous files" on page 49. **2** Click the Icon field, and then click the ... button that appears. **3** In the Open File dialog box, locate and select the image file (.jpg, .png, .bmp, or .ico), and then click Open. 
Caption	To change the label that appears on the button, type the new text in the field.
Font size	*Optional*. To change the font size for the selected control and subsequently created controls, type a new number in the field. Default: 10
Font color, Background color (button)	*Optional*. To change the font or background color for the selected control and subsequently created controls: **1** Click the field, and then click the ▼ that appears. **2** In the palette list, select the color, or click Custom Color to open the Select Color dialog box and create a custom color. Default font color: Web Black Default background color: 240, 240, 240

Property	Description
Minimum user level to interact	Select the VWorks access level required for this form control. The options are Operator (default), Technician, or Administrator. For example, you would probably set this property to Operator for a Run Protocol button. But you might want to set this property to Technician for a Menu Action button.

Form button properties

In addition to the properties common to specialized buttons, the Form button has the following properties:

Property	Description
Form file	To select the form file that this button will open: 1 Click the field, and then click the  button that appears. 2 In the Open File dialog box, locate and select the .VWform file, and then click Open.
Form switch	Select the check box if you want the currently open form to close when the user clicks the Form button. Clear the check box if you want both forms to remain open.

Menu Action button properties

IMPORTANT

The Menu Action button may give a VWorks operator the ability to perform actions that normally require technician-level access. If you want to prevent this, set the **Minimum user level to interact** property to Technician or Administrator.

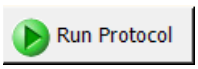
In addition to the properties common to specialized buttons, the Menu Action button has the following property:

Property	Description
Action	Click the field, and then click the  that appears. Click the command that you want the button to perform.

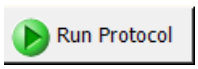
The following table lists the Action commands that you can select from.

Menu Action button commands	For details, see...
• Display Gantt Charts • Labware Editor • Liquid Library Editor • Pipette Technique Editor • System State Editor • Tip State Editor • Configure Labware • Backup, Restore (Backup Manager) • Log Management • Error Library • Inventory Editor • Global Options (Tools > Options)	These commands are also available in the Tools menu of the VWorks window. For a description of the Tools menu commands, see "Tools menu" on page 748.
• Export (exports the form and all associated files)	The Export command is also available in the File menu. For details on the Export command, see "Exporting and importing protocols and associated records" on page 703.
• Hit Picking Format wizard–Bravo	This command is also available in the Tools menu of the VWorks window. For a description of the Tools menu commands, see "Tools menu" on page 748.
• Toggle Watcher	This command is also available in the Tools menu as Watcher is OFF, Watcher is ON. For details, see "Setting up and using the Watcher tool" on page 755.
• Diagnostics	Opens device diagnostics for the loaded device file. This command is also available in the Control toolbar.
• Experiments Editor • Experiments Archive • Add Experiment Note	*Experiments database only*. See "Menu Action buttons for the Experiments Editor" on page 257.

Run Protocol button properties

The Run Protocol button 

opens the Run Configuration Wizard for the protocol. This button is available only if the protocol is open and no run is currently in progress.

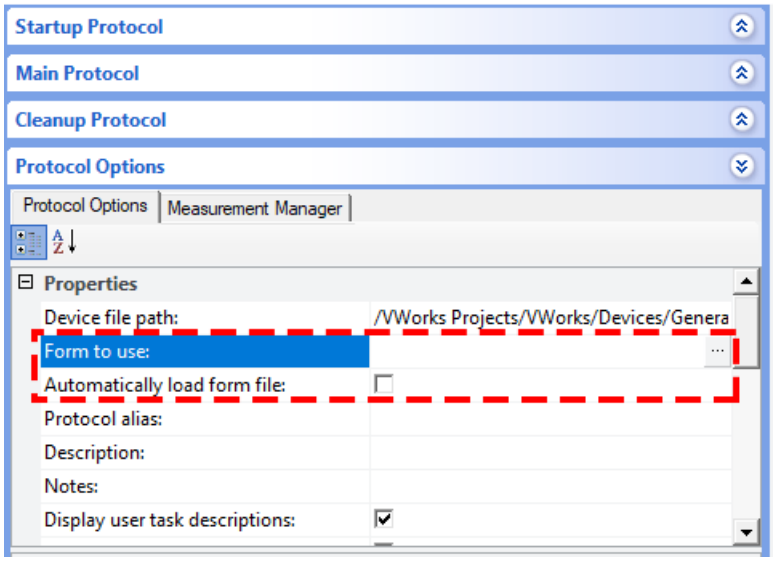
In addition to setting the properties common to specialized buttons, use the following procedure for the Run Protocol button 

to link the form to the protocol.

To link the form to the protocol:

1 In the VWorks window, open the protocol, and click **Protocol Options.**

2 Under **Properties**, set the following properties.



a Click the **Form to use** field, and then click the ... button that appears.

b In the **Open File** dialog box, locate and select the form file that you are creating. (.VWForm), and then click **Open**.

c In the **Automatically load form file** field:

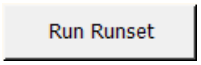
- Ensure the check box is selected (default) if the form should open automatically anytime the protocol is opened.
- Clear the check box if the form should not open automatically when the protocol is opened.

Run Runset button properties

The Run Runset button [Run Runset] starts the specified runset. The Run Configuration Wizard does not appear in this case.

In addition to the properties common to specialized buttons, the Run Runset button [Run Runset] has the following property:

Property	Description
Runset file	To select the runset that this form will run: 1 Click the field, and then click the ... button that appears. 2 In the Open File dialog box, locate and select the .rst file, and then click Open.

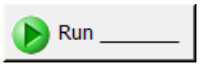
When configuring the Run Runset button 

, make sure you select the **Use global context for variables** check box in the **Form Designer** window.

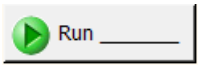
If you want the JavaScript variables used by the form controls to interact with variables in the protocols, ensure that the **Use global context for this protocol** check box is selected in the **Protocol Options** for each protocol.

If one of the protocols uses its own private context (the **Use global context for this protocol** check box is not selected), you can include a `getGlobalObject()` JavaScript call in each case where a variable in the protocol must read from, or write to, one of the variables used by the form. For details, see "VWorks-defined functions" on page 103.

Run Specified Protocol button properties

The Run Specified Protocol button 

starts the run for the specified protocol immediately without displaying the Run Configuration Wizard, unless the button properties specify a script for prerun checks. If a script for prerun checks is specified, the checks are performed before the protocol runs. In this case, the button is grayed the instant it is pressed, and remains grayed until either the checks fail or the protocol completes.

In addition to the properties common to specialized buttons, the Run Specified Protocol button 

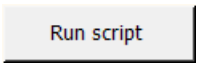
has the following properties:

Property	Description
Protocol file	To select the protocol that this form will run: 1 Click the field, and then click the  button that appears. 2 In the Open File dialog box, locate and select the .pro file, and then click Open.
Number of times to run	Type the number of times the protocol will run after the form user clicks the run button.

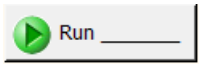
Property	Description
Script for pre-run checks	Specify a script only if you want to perform prerun checks before the protocol starts. To enter the script that this button will run: 1 Click the field, and then click the  button that appears. 2 In the Input Text dialog box, type the JavaScript code in the box. Alternatively, click Browse. In the Open File dialog box, select a file that contains the JavaScript code, and then click Open. 3 Click OK to close the Input Text dialog box. 4 Specify the JavaScript variable for pre-run check property.

JavaScript variable for pre-run check	Type the name of the variable that contains the result of the prerun checks specified in the Script for pre-run checks property.
	For example, if this variable name is goOrNoGo, the script might be as follows:
	`\var goOrNoGo = false;`
	`if (all checks pass)`
	`goOrNoGo = true;`
	`else`
	`Notify user of failure.`
	When the Run Specified Protocol button is pressed, the prerun checks would be performed. If the checks failed, the form's user would be notified of the failure. If the checks passed, the protocol would start.

Run Specified Script button properties

The Run Specified Script button 

runs the specified JavaScript code.

Alternatively, you can specify a script for prerun checks in the Run Specified Protocol button 

properties.

In addition to the properties common to specialized buttons, the Run Specified Script button has the following property:

Property	Description
Script	To enter the script that this button will run: 1 Click the field, and then click the  button that appears. 2 In the Input Text dialog box, type the JavaScript code in the box. Alternatively, click Browse. In the Open File dialog box, select a file that contains the JavaScript code, and then click Open. 3 Click OK to close the Input Text dialog box.

Related information

For information about...	See...
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
Forms and Form Designer overview	"About forms for running protocols" on page 178
Controls under General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Controls under Specialized Display Controls in the Form Designer window	"Adding indicators for elapsed time and progress to a form" on page 202
Form properties in the Form Designer window	"Setting the form properties" on page 240
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22

Adding indicators for elapsed time and progress to a form

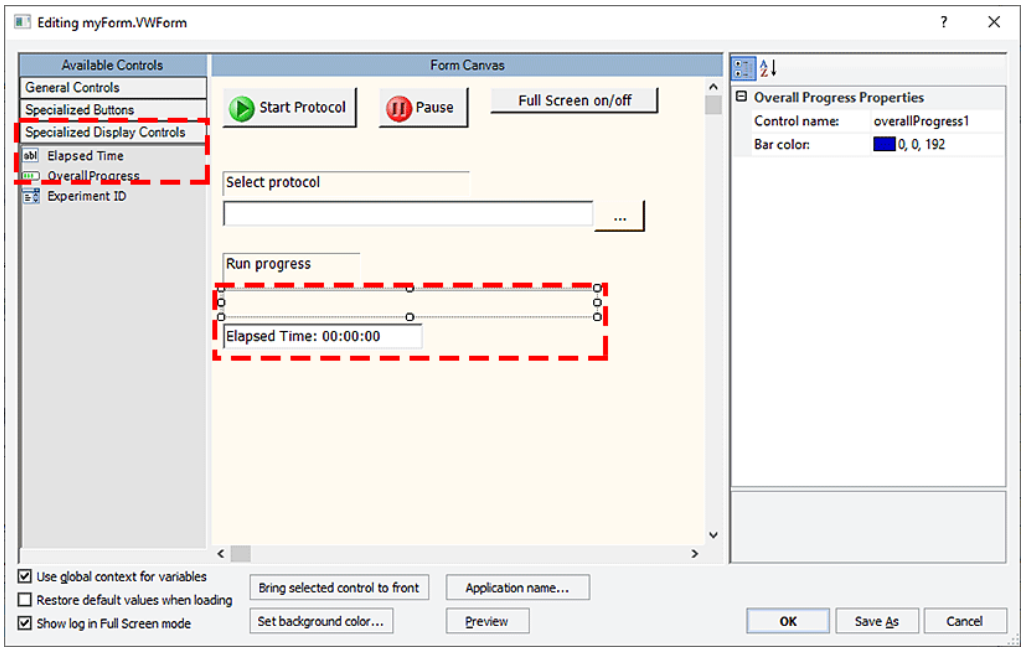
About this topic

To create or edit forms, you must be an administrator or technician. You should also have an understanding of how to create protocols in the VWorks software.

This topic describes how to configure the Elapsed Time and Overall Progress controls in the Specialized Display Controls area of the Form Designer window.

For details on the Experiment ID control, see "Configuring a form for experiment IDs and Experiments database" on page 249.

Figure Form Designer window and the Specialized Display Controls



For more details on creating forms, see "Workflow for creating or editing a form" on page 181.

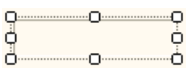
Procedure

To add indicators for elapsed time and progress:

1 In the **Form Designer** window under **Available Controls**, click **Specialized display controls**, and then drag the desired control to the **Form Canvas** area.

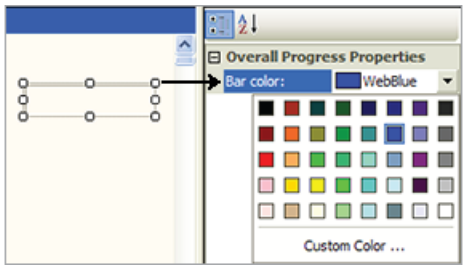
An invisible autogrid assists you in aligning the control in the form. To turn off the autogrid, press ALT while dragging the control into position.

2 In the **Form Canvas** area, click the control so that a resizing box appears around the border.



To resize the control, drag one of the sizing handles.

3 Edit the control properties as required. The following table describes the properties.

Property	Description
Control name	The descriptive name associated with the control. The Control name does not appear on the form. Use a functionally descriptive name so that references to this control are recognizable in the Main Log and in audit trail reports. For details, see "About the naming of form controls" on page 184.
Font size	*Elapsed time only*. To change the font size, type a new number in the field. Any controls that you create subsequently will use the new font size. Default: 10
Font color or Bar color	*Optional*. To change the color for the selected control and subsequently created controls, click the field, and then click the ▾ that appears. Click the color in the palette list, or click Custom Color to open the Select Color dialog box and create a custom color. 

Note: To create a caption for the Overall Progress control, you can combine the Overall Progress control with the Static text control. See "Overview of the General Controls" on page 205.

Note: To have one control overlap another control on the form, select the control that you want on top, and then click **Bring selected control to front**.

Related information

For information about...	See...
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
Controls under General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Controls under Specialized Buttons in the Form Designer window	"Overview of the specialized buttons" on page 188

For information about...	See...
Form properties in the Form Designer window	"Setting the form properties" on page 240
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22

Adding form controls that allow editing or runtime data display

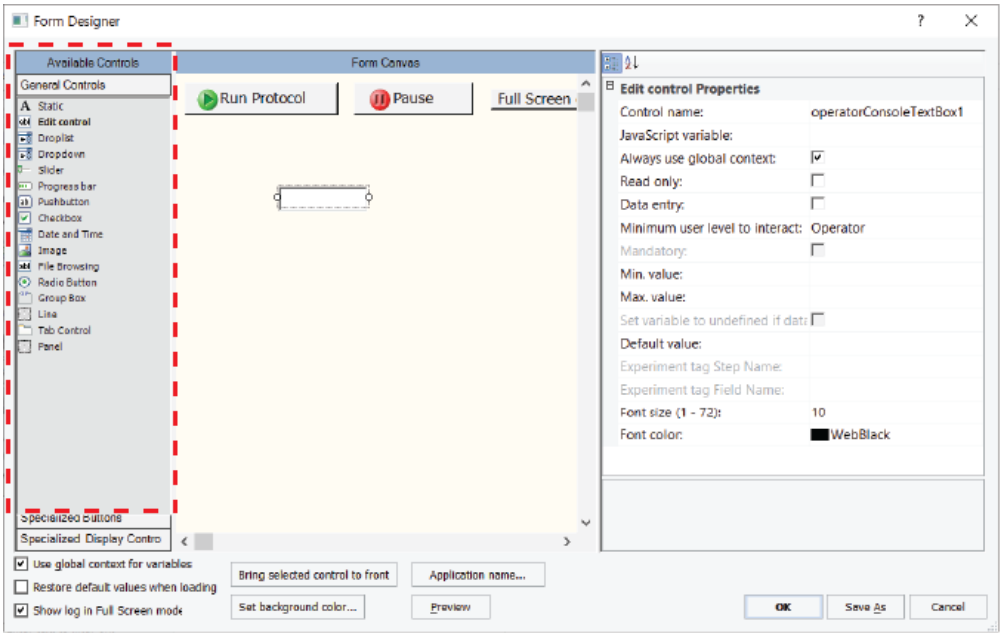
About this topic

To create or edit forms, you must be an administrator or technician. You should also have an understanding of how to create protocols and define JavaScript variables in the VWorks software.

This topic describes the General Controls in the Form Designer window and how to configure them.

- "Overview of the General Controls" on page 205
- "Configuring the General Controls" on page 210

Figure General Controls in Form Designer window

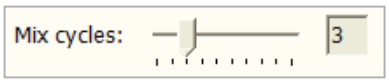
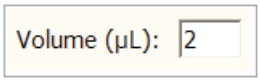


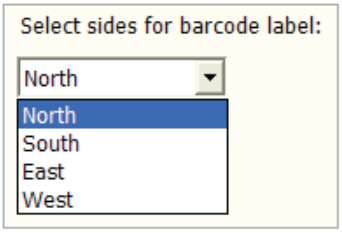
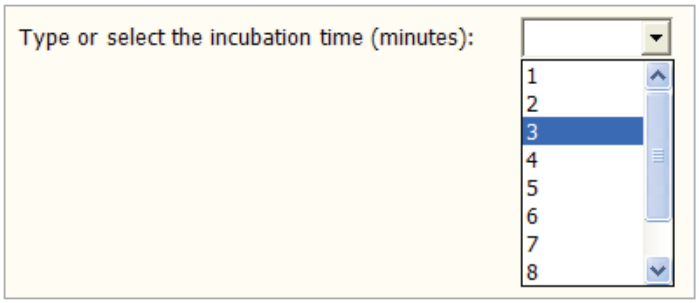
For more details on creating forms, see "Workflow for creating or editing a form" on page 181.

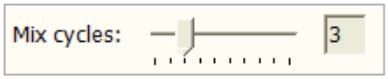
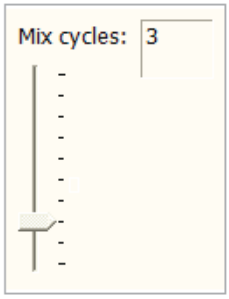
Overview of the General Controls

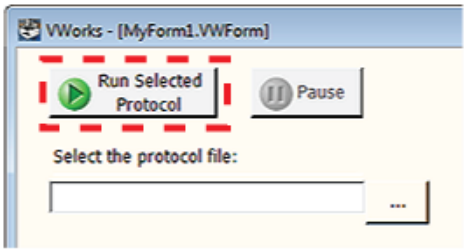
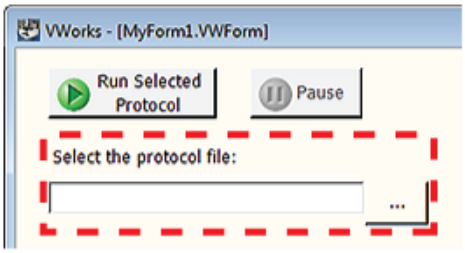
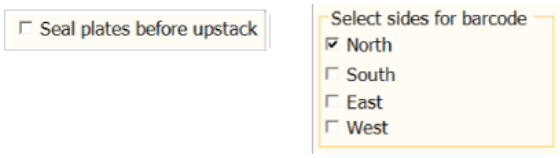
The controls available under General Controls can be assigned JavaScript variables to allow users to edit data or to display runtime data on the form. For example, you might want the form to display the volume dispensed or to allow the form's user to enter the number of mix cycles. The JavaScript variable assigned to the control must also be assigned to the given task parameter value.

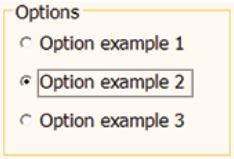
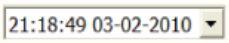
The general controls include the following:

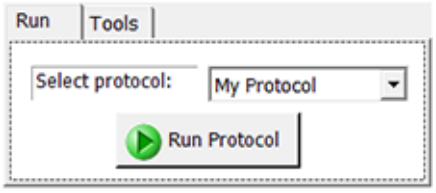
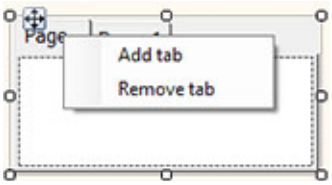
Control	Description
Static text	Displays read-only text. Use this control to create: • Captions for other controls on the form, such as drop-down lists, progress bars, sliders, and edit controls. • Read-only displays for runtime data. The JavaScript variable must be assigned in the Static control properties and in the corresponding task parameter in the protocol. The following figure shows an example that consists of two Static text controls in combination with a Slider control. A Static text control (left) provides the caption, *Mix cycles,* and a Static text control (right) displays the value of the current slider setting. 
Edit control	Allows form users to enter or edit data. The JavaScript variable must be assigned in the Edit control properties and in the corresponding task parameter in the protocol. The following figure shows an Edit control (right) that allows the form users to edit the volume. This control has an assigned JavaScript variable (`vol`), which is also assigned in the corresponding task parameter in the protocol.  *Note:* A Static text control (left) provides the caption, *Volume (uL)*.

Control	Description
Droplist	Provides a drop-down list of choices for a task parameter. In the following figure the Droplist control has an assigned JavaScript variable (`BCside`) that is also specified in the protocol task parameter for defining the barcode side of the labware.  *Note:* A Static text control provides the caption for the Droplist control.
Dropdown controls	Provides a drop-down list of choices and allows users to type arbitrary text into the edit field. The Dropdown control in the figure has an assigned JavaScript variable (`time`) that is also specified in the protocol task parameter for defining the incubation time.  *Note:* A Static text control provides the caption for the Dropdown control.

Control	Description
Slider	Enables users to set a value on a continuous range of possible values. The following figure shows a horizontal and vertical slider, where the position of the slider is coordinated with a Static text control (left) or Edit control (right).   In the horizontal example (left), the user moves the slider to set the Mix task parameter value, while the value updates in the display. In the vertical example (right), the user can type a number in the box or move the slider to set the Mix task parameter value.
Progress bar	Provides a visual indication of task progress. The JavaScript variable must be assigned in the control properties and in the corresponding task parameter in the protocol. In the following figure, the gray bar shows how much of the task has been completed.  *Note:* A Static text control provides the caption, *Progress*. *Note:* To display the progress for the entire protocol or runset that the form is running, use the Overall Progress bar under Specialized Display Controls.

Control	Description
Pushbutton	Allows you to write your own script for the action that the button will perform. The following figure shows a form with a custom pushbutton for running a protocol that the form's user selects in a File Browsing control. The script specified in the Pushbutton Properties calls the file path defined as the JavaScript variable for the File Browsing control.  *Note:* For more details on how to set up this example, see "Example: Creating a scripted Pushbutton control in a form" on page 232.
File Browsing	Allows users to select or save the specified file type.  A Static text control provides the caption, *Select the protocol file*.
Checkbox	Allows users to enable or disable the corresponding actions. The JavaScript variable must be assigned in the Checkbox control properties and in the corresponding task parameter in the protocol. The following figure shows two examples.  If the user selects the "Checkbox properties" on page 228, the assigned JavaScript variable is set to 1. If the user clears the check box, the assigned JavaScript variable is set to 0.

Control	Description
Radio button	Presents choices from among several items. Each Radio button control has a JavaScript variable that is also assigned in the corresponding protocol task parameter.  If a user selects an option, the assigned JavaScript variable is set to 1. If the user clears the option, the assigned JavaScript variable is set to 0.
Date and Time	Displays a preset date and time or a date that the user can set. To correspond with a given task, the JavaScript variable must be assigned in the control properties and in the corresponding task parameter in the protocol.  *Note:* You can use the Elapsed Time indicator for the entire run. See "Adding indicators for elapsed time and progress to a form" on page 202.
Group box	Displays a rectangle that can be resized and labeled to visually group other controls together. To correspond with a given task, the JavaScript variable must be assigned in the control properties and in the corresponding task parameter in the protocol.
Image	Allows you to place a static image on the form. To correspond with a given task, the JavaScript variable must be assigned in the control properties and in the corresponding task parameter in the protocol.
Line	Provides a vertical or horizontal line that you can use to visually separate items on the form.

Control	Description
Tab Control	Displays tabbed pages in the form. You can add different controls and information in each tabbed page. ***Figure*** Tabbed pages  By default, when you drag a Tab Control onto the Form Canvas, the software displays two tabbed pages. To add another tabbed page, right-click the tab area, and select **Add tab**. To remove a tabbed page, right-click the tab, and then select **Remove tab**. To change the tab names, see "Tab Control properties" on page 231. ***Figure*** Adding or removing tabbed pages 
Panel	Displays a container in which you can add controls and information. You can add one or more panels in a form to group or categorize different controls. ***Figure*** Panel containing different controls 

Configuring the General Controls

To add a control to the form:

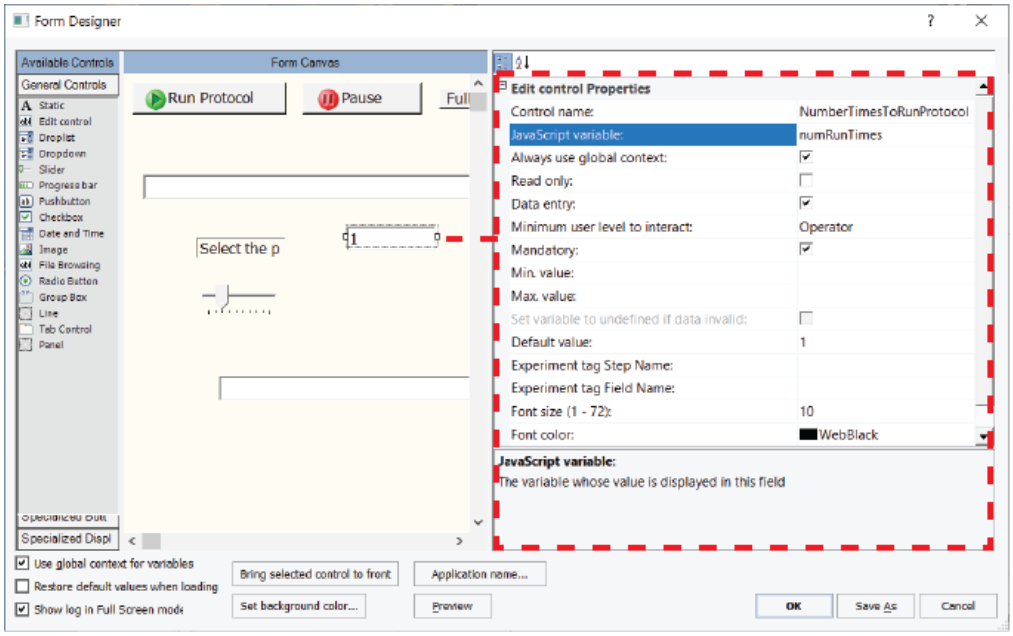
1 Under **Available Controls**, click **General Controls**, and then drag the desired control to the **Form Canvas** area.

Note: To turn off the autogrid, press ALT while dragging the control into position.

2 In the **Form Canvas** area, click the control that you want to set up, so that a resizing box appears around the border. To resize the control, drag one of the sizing handles.

Note that the corresponding properties appear for the selected control as the following figure shows.

Figure Form Designer window displaying properties for the selected control



Note: Click **Preview** to display the image that the form's user will see.

3 Edit the **Properties** for the control. The properties can include a combination of the following depending on the control:

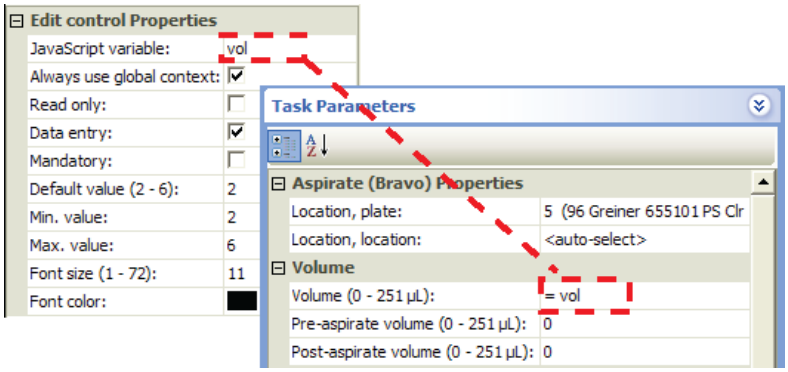
- "Properties common to the general controls" on page 212
- "Static text control properties" on page 216
- "Edit control properties" on page 217
- "Droplist and Dropdown properties" on page 218
- "Slider properties" on page 219
- "Progress bar properties" on page 221
- "Pushbutton control properties" on page 222
- "File browsing control and File Browse button properties" on page 223
- "Checkbox properties" on page 228
- "Radio Button properties" on page 229
- "Image (static) control properties" on page 230
- "Date and Time properties" on page 230
- "Group Box properties" on page 230
- "Tab Control properties" on page 231
- "Panel properties" on page 231

The Line control does have any properties that you can set.

Note: If you want one control to overlap another control on the form, select the control that you want on top, and then click **Bring selected control to front**.

Properties common to the general controls

The following table lists the properties for all controls. A given control might use only a subset of the properties listed.

Property	Description
Control name	The descriptive name associated with the control. The Control name does not appear on the form. Use a functionally descriptive name so that references to this control are recognizable in the Main Log and in audit trail reports. For details, see "About the naming of form controls" on page 184.
JavaScript variable	The JavaScript variable that will be associated with the focused control. Type any string that qualifies as valid syntax for a JavaScript variable name. In the protocol, type the variable name for the corresponding Task Parameters property. For example, if you want the form's user to enter the aspirate volume for a given task, you could use an Edit control that has a JavaScript variable of `vol`. In the protocol, you would specify the Aspirate Properties for Volume as = `vol`.  For more details on using JavaScript variables in the Task Parameters area, see "Using simple variables" on page 93.

Property	Description
Always use global context	If the form's **Use global context for variables** check box is selected, you can ignore this setting. If the form's **Use global context for variables** check box is not selected, specify which context to use for the control's JavaScript variable: • *Global context*. Select this check box. For example, if the variable in question will be acted on before the protocol is running or is scheduled to run, use the global context. **IMPORTANT** Any protocol that uses the global context and has JavaScript variables of the same name will be affected by changes you make to this variable, even if this form's **Use global context for variables** check box is not selected. • *Context of the running protocol*. Clear this check box. For example, if the variable in question will communicate with the protocol once the protocol is running or is scheduled to run, you may not need to use the global context. For more details on JavaScript context, see "Understanding JavaScript context in form design" on page 243.
Read only	The state that determines whether the form's user can edit the contents of the control. • Select the check box to prevent users from editing the control contents. • Clear the check box to allow users to edit the control contents.
BorderStyle	The border surrounding the control. You can specify one of the following border styles: • *FixedSingle*. Adds a single line that encloses the control. • *Fixed3D*. Adds a 3-dimensional effect to the border's appearance.
BackgroundImage	The image displayed as the background for the control. The selected image is displayed on top of the selected background color. Image files must be imported into the VWorks file storage system before they are available for use in the VWorks software. To import an image file, see "Importing and exporting miscellaneous files" on page 49. To use the image in the form, click the BackgroundImage field, and then click the  button that appears. In the Open File dialog box, locate and select the image file (.jpg, .png, .bmp, or .ico), and then click Open.

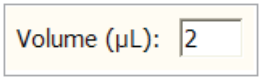
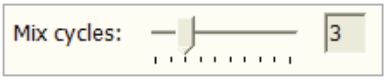
Property	Description
BackgroundImageLayout	How the selected image is displayed. Select one of the following display methods: • *Center*. Center-aligns the image on the control. • *Stretch*. Stretches the image to fit the size of the control. The dimensional ratio of the image is not maintained. • *Tile*. Uses multiple copies of the image to cover the entire control area. The images are in their original sizes and are placed adjacent to one another. • *Zoom*. Magnifies the image such that the width or length covers the width or length of the control. The dimensional ratio of the image is maintained.
Background_Color	The color of the control. Click the field, and then select the desired color.
Data entry	The option to allow the user to edit the value in the control when the protocol is paused. • Select the check box to allow form users to edit the value when a protocol is not running and when a running protocol is paused. • Clear the check box to allow users to edit the value only when a protocol is not running. For the data entry controls, you can also provide the Print, Reset, and Save data entry buttons from the Specialized Buttons area. For details, see "Overview of the specialized buttons" on page 188. *Note:* The Data entry property is not available if the Read only check box is selected.
Allow data entry when paused	The option to allow the control to be editable while a running protocol is associated with the form. • Select the check box (true) to allow the control to be editable while a running protocol is associated with the form. This parameter defaults to true when the control is created, or when opening a form in the Form Editor that was created in a version earlier than VWorks 14.1.1. • Clear the check box if you want the control to be editable only when the protocol associated with the form is not running.
Minimum user level to interact	The VWorks access level required for this form control. The options are Operator (default), Technician, or Administrator.
Mandatory	The option to require the form user to enter a value for this field before starting the associated protocol or runset. Select the check box to require the user to enter a value for the field. Clear the check box if the user can start the protocol or runset without entering a value for the field. Available if the Data entry check box is selected.

Property	Description
Default value	The value the form should display when the user resets the field. Type the default value in the box. Special conditions apply for some controls, such as check boxes and radio buttons. For details, see the property description for the specific control.
Min value, Max value	A range of permitted values. The value saved to the associated JavaScript variable will be a number. Type the minimum and maximum value to define the range of values. If the form's user types a value outside this range, an error message appears. To allow the user to type any value in an Edit control, leave the Min and Max value properties blank. The value saved to the associated JavaScript variable will be a number, unless the value appears to be something other than a number. In which case, the value will be saved as a string. Special conditions apply for some controls, such as sliders. For details, see the property description for the specific control.
Experiment tag Step Name	*Experiments database only*. Specifies the name used to create an experiment tag (Step Name) in the Experiments database for the form control when a protocol is run using this form. For details, see "Configuring a form for experiment IDs and Experiments database" on page 249. *Note:* The Experiment tag Step Name property is available only if the Data entry check box is selected.
Experiment tag Field Name	*Experiments database only*. Specifies the name to be used for the corresponding experiment tag in the Experiments database when a protocol is run using this form. For details, see "Adding form controls that allow editing or runtime data display" on page 204. *Note:* The Experiment tag Field Name property is available only if the Data entry check box is selected.
Order in Report	*Experiments database only*. Specifies the order in which the associated data tags for this control appear in an experiments report relative to other controls in the same step. For details on creating forms to track experiment data, see "Configuring a form for experiment IDs and Experiments database" on page 249.

Property	Description
Dock	The option to anchor the control to a position on the Form Canvas. Select one of the following positions: • *None*. You can move the control freely on the canvas. Select None if you want to manually position the control, and you prefer to set the size of the control using the Width and Height properties. • *Top*. The top side of the control is adjacent to the top of the Form Canvas. The control's width is increased to match the width of the canvas. Although the width is fixed, you can change the Height value. • *Left*. The left side of the control is adjacent to the left side of the Form Canvas. The control's height is increased to match the height of the canvas. Although the height is fixed, you can change the Width value. • *Bottom*. The bottom side of the control is adjacent to the bottom of the Form Canvas. The control's width is increased to match the width of the canvas. Although the width is fixed, you can change the Height value. • *Right*. The right side of the control is adjacent to the right side of the Form Canvas. The control's height is increased to match the height of the canvas. Although the height is fixed, you can change the Width value. • *Fill*. The control's size is increased so that it completely covers the Form Canvas area. The width and height of the control are fixed.
Font size	The font size of the text that appears in the control. To change the font size, type a new number in the field. Any controls that you create subsequently will use the new font size. Default: 10
Font color	The color of the text that appears in the control. To change the color for the selected control and subsequently created controls, click the field, and then click the ▾ that appears. Click the color in the palette list, or click Custom Color to open the Select Color dialog box and create a custom color. Default font color: Web Black Default background color: 240, 240, 240

Static text control properties

The Static text control has the following properties in addition to the properties common to the other general controls.

Property	Description
Display string	Type the text that you want to appear on the form, for example, a caption for another control on your form. The following figure shows an example of a Static text control with a display string that provides the label, *Volume (uL)* for an Edit control.  *Note:* Display string is enabled only if the JavaScript variable property is blank. *Note:* To use the symbol µ, copy the characters to the Microsoft Clipboard from other software. Right-click the field and choose Paste to insert the character into the field.
Border	To add a border around the control, ensure the check box is selected (default). For example, a border is useful if you want to provide a display of runtime data. To remove the border, clear the check box. For example, if you are creating a caption for one of the other controls, a border is unnecessary. The following figure shows an example of two Static text controls combined with a Slider control. The caption on the left is a Static control with no border. The display on the right is a Static control with a border. 

To coordinate the Static text control with a protocol task parameter, both the control and the task parameter must use the same JavaScript variable.

To coordinate the Static text control with another control, for example, the slider, both controls must have the same assigned JavaScript variable and both controls must specify the global JavaScript context so that they can communicate with each other.

Edit control properties

The Edit control has the following property in addition to the properties common to the other general controls.

Property	Description

Set variable to undefined if data invalid	The option to automatically assign a JavaScript value as *undefined* in a case where a variable that requires a numeric value has been declared, but it has not been assigned a numeric value.
	• If you select this option and the corresponding field requires a numeric value, but the field is left blank or contains a string instead of a number, the software executes the following internal script:
	`var <variable_name> = undefined;`
	Protocol scripts can test for this undefined value and act accordingly.
	• If you do not select this option and you have a variable that requires a numeric value, the control can yield an unexpected result if a valid numeric value is missing.
	For example, suppose you specified the following properties for an Edit control:
	– Data entry: checked
	– Mandatory: unchecked
	– JavaScript variable: vol
	– Always use global context: checked
	– Min value: 1, Max value: 251
	If the form user enters a value of 55 for the `vol` variable and then runs the associated protocol, the software executes the following internal script:
	`var vol = 55;`
	But suppose that for a subsequent run, the user leaves this field on the form blank. This time the software executes the internal script:
	`var vol;`
	This script ensures that the variable exists but does not assign a value to it. If the global context is not cleared in this case, the scripts in the protocol will behave as though the user entered 55 on the form, even if this was not the intended value.

Always treat as string	Select the check box if you want to ensure that a value will always be treated as a string, even if it looks like a number. Clear the check box if you want the software to treat the entered text as a number if it looks like a number, and treat it as a string if it does not look like a number.

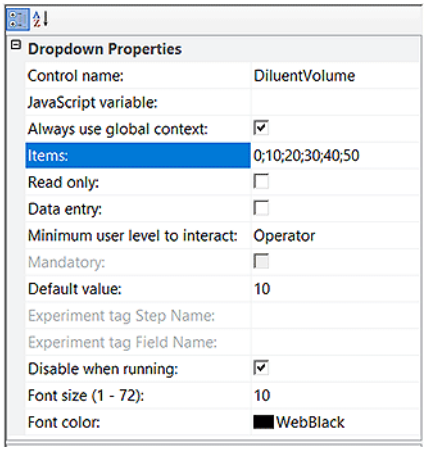
Droplist and Dropdown properties

The Droplist and Dropdown controls have the following properties in addition to the properties common to the other general controls.

Note: You might want to use these controls in combination with a Static text control to display a caption for your drop-down list or drop-down combo box.

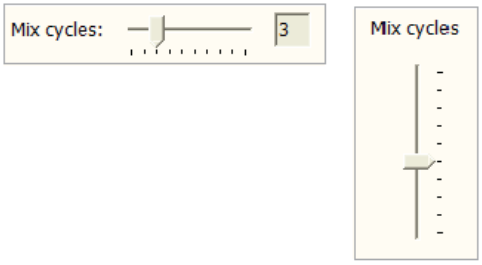
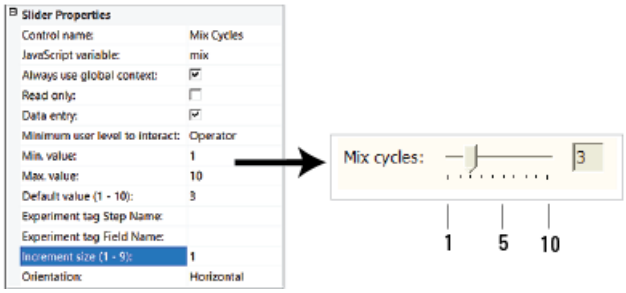
Property	Description
Items	Type the list of items that will appear in the list of the Droplist or Dropdown control, where each item is separated by a semicolon with no spaces. For example, type 3;6;9;12 to provide the form's user with the choices 3, 6, 9, 12.
Default value	Type the item from Items list that will display in the form by default. **IMPORTANT** Make sure that the default value is also in the Items list. Otherwise, the Droplist control will display the first list item by default. The Dropdown control will display the default value even if the value is not included in the Items list.
Disable when running	Select the check box to prevent the operator from changing the selected Dropdown item while the protocol is running. Clear the check box to enable the operator to change the selected Dropdown item during the protocol run time. This property is available only when the Read only check box is cleared. If the Read only property is selected, the Disable when running property is not available.

Figure Dropdown Properties example



Slider properties

The Slider control has the following properties in addition to the properties common to the other general controls.

Property	Description
Orientation	Click the field to display the ▾ and then click one of the following: • *Horizontal*. The scale reads from left to right, Min value to Max value. • *Vertical*. The scale reads from bottom to top, Min value to Max value. The figure shows an example of a horizontal and vertical slider. The horizontal slider is combined with an Edit control that displays the changing numeric value as the slider is moved. 
Min value, Max value	To specify the range of values that the slider represents, type a minimum and maximum numeric value. The slider has a tick mark for each numeric value in the range. For example, a slider with a range of 1–10 has 10 tick marks, and a slider with a range of 1–20 has 20 tick marks.  When the form's user moves the slider, the resulting value is written to the control's JavaScript variable as soon as the slider is released.
Increment size	Type a positive number to use as the increment that the numeric value changes as the slider is moved. For example, if the slider's range is 1–10 and the increment size is 1, the value will change by 1 each time you move the slider. If the increment is 2, the value will change by 2 each time you move the slider. The increment size should be in relation to the slider's range of values. For example, a slider that has a narrow range of values typically would not have a large increment size. Likewise, a small increment size may be insufficient for a slider that has a very wide range of values.

You can use the Slider control in combination with the:

- Static text control to display a caption for the slider or display the current value of the slider's setting.
- Edit control to display the current value of the slider's setting or allow users to enter a value instead of using the slider.

To coordinate the slider setting with a Static text control or Edit control and the protocol task parameter setting:

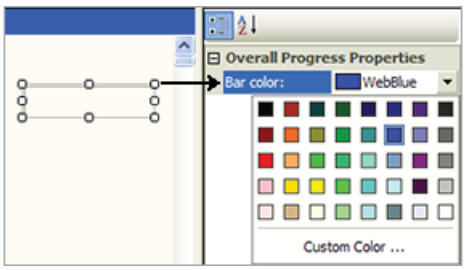
- Both the Slider and Static or Edit controls must have the same JavaScript variable assigned (for example, `mix`).
- The variable assigned in the Slider and Static or Edit controls must be in the global context so that the two controls can communicate.
- The corresponding Mix task parameter in the protocol must also have the same JavaScript variable assigned (for example, `mix`).

If the slider's Data Entry property is not selected, the slider will move according to the current value of the underlying JavaScript variable. The behavior of the slider is undefined if the JavaScript variable does not have a numeric value, or if the variable's value is outside the slider's range.

Progress bar properties

The Progress bar control has the following properties in addition to the properties common to the other general controls.

Note: You might want to use this control in combination with a Static text control to create the progress bar caption.

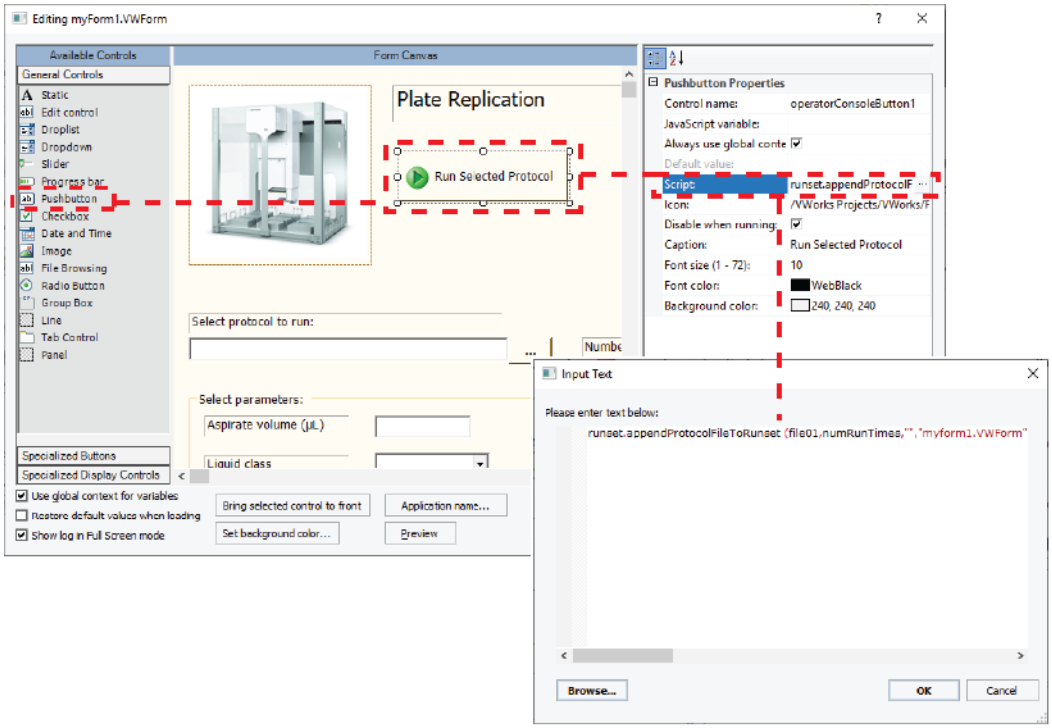
Property	Description
JavaScript variable	Must have a numeric value that is a positive number and the value must not be greater than the Max value setting.
Max value	Type a numeric value larger than 0. The value represented by the left end of the progress bar is always 0. The Max value determines the value represented by the right end of the progress bar.
Bar color	*Optional*. To change the color, click the field, and then click the  that appears. Click the color in the palette list, or click Custom Color to open the Select Color dialog box and create a custom color. 

The progress bar moves according to the current value of the underlying JavaScript variable. All progress bars are horizontal.

Pushbutton control properties

The Pushbutton control can execute any JavaScript code that you specify. You provide the script in the Pushbutton Properties, as the following figure shows.

Figure Workflow to create a pushbutton



In addition to the properties common to the general controls, the Pushbutton control has the following properties. For an example of how to set up a pushbutton using a script, see "Example: Creating a scripted Pushbutton control in a form" on page 232.

Property	Description
JavaScript variable	The JavaScript variable that will be associated with the control. Type any string that qualifies as valid syntax for a JavaScript variable name. The JavaScript variable can be used to exchange a result or value between the protocol and the form. A variable can also be shared between different areas of the same form, for example, to update another field within a form with a button. If you provide a variable, you can specify the Default value for the JavaScript variable.

Property	Description
Always use global context	Specify the context in which to execute the script: • *Global context*. Select the check box to use the global context. For example, if the button is to be pressed while no protocol is running, you must use the global context. **IMPORTANT** Any protocol that uses the global context and contains JavaScript variables of the same name will be affected by changes you make to this variable, even if this form's **Use global context for variables** check box is not selected. • *Context of the running protocol*. Clear the check box to use the protocol's private context.
Default value	Specify the default value of the JavaScript variable.
Script	Click the field, and then click the  button that appears. In the Input Text dialog box, type the JavaScript code to be executed when the pushbutton is pressed, or click Browse to select the file (.txt) that contains the script. *Note:* If the script specifies a file on your computer, and you later export the form with this pushbutton, you must specify the file in the Additional Files page of the Export Wizard.
Icon	*Optional*. You may select an image to use as an icon. Image files must be imported into the VWorks file storage system before they are available for use in the VWorks software. To import an image file, see "Importing and exporting miscellaneous files" on page 49. To use the image in the form, click the Icon field, and then click the  button that appears. In the Open File dialog box, locate and select the image file (.jpg, .png, .bmp, or .ico), and then click Open.
Disable when running	To make the button unavailable (grayed) whenever the form is running a protocol or runset, select this check box (default). To make the button enabled at all times, clear this check box.
Caption	To change the label that appears on the button, type the new text in the field.

File browsing control and File Browse button properties

The Form Designer window includes the following file browsing controls under Available Controls:

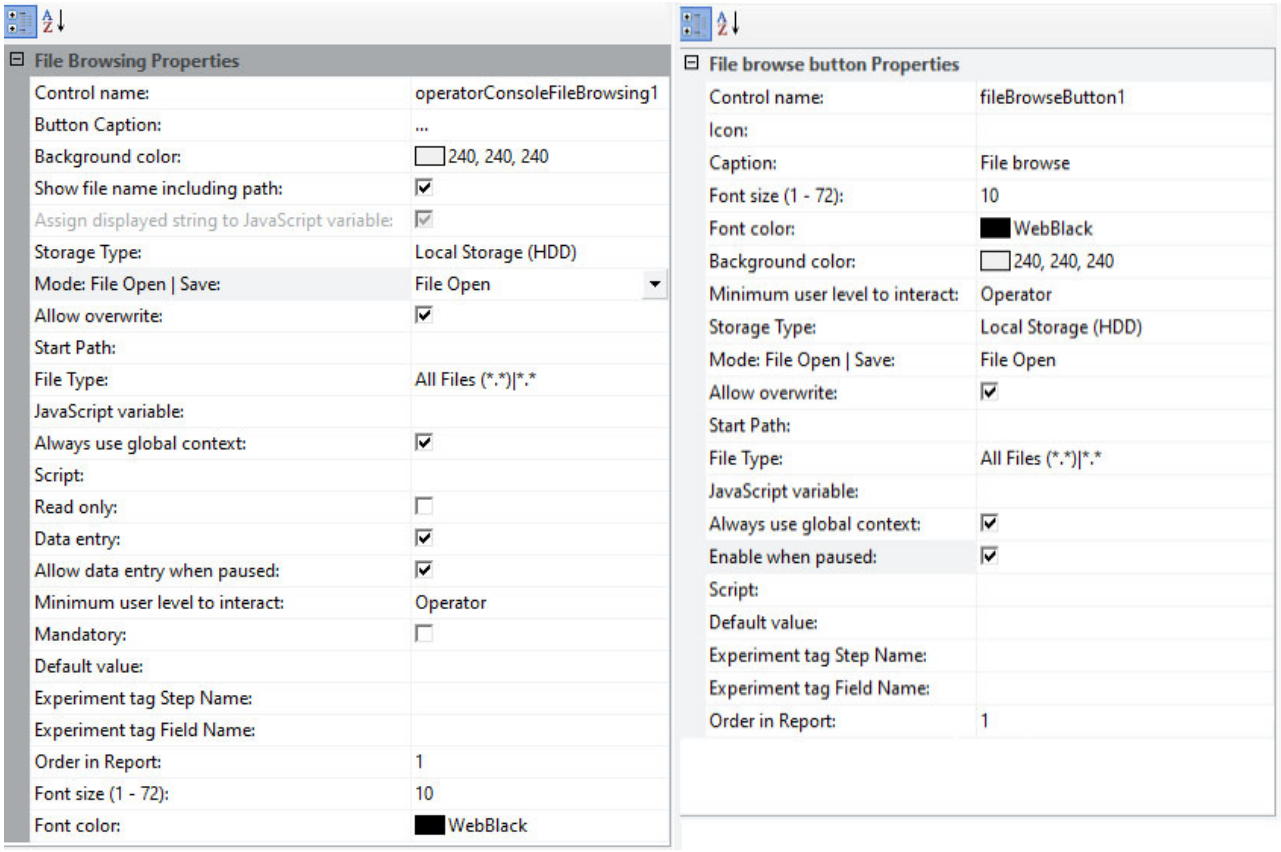
- **File Browsing control (General Controls)**. Enables the form's user to select or save a file at a specified location and displays the selected file name.



- **File Browse button (Specialized Buttons)**. Enables the form's user to select or save a file at a specified location, but it does not include a file name display.



Figure Properties for the File Browsing control (left) and the File Browse button (right)



The File Browsing control and File Browse button have the following properties in common:

Property	Description
Control name	See "Properties common to the general controls" on page 212 for a description.
Storage Type	The type of Shared Services storage. The options are: • Shared Services Repository — Secure Storage • Local Storage (HDD) — local file system
Mode: File Open | Save	The type of dialog box to open. The options are: • File Open • File Save As

Property	Description
Allow overwrite	*Shared Services Repository storage only*. Specifies whether to allow overwriting of a file that already exists in Secure Storage: • Select the check box to allow overwriting. • Clear the check box to prevent overwriting.
Start Path	The file path to display in the File Open or File Save As dialog box, for example, /VWorks Projects/VWorks/AM Methods.
File Type	The filter to use for the available types of files, for example, text (*.txt) or (*.csv), Excel (*.xlsx) or (*.xls), method files (*.mth), and so forth.
Script	The JavaScript that this File Browse button or File Browsing control will execute. The script will run after the form's user presses the button, selects or enters a file name, and then clicks OK. The JavaScript variable's value will hold the selected full path file name. ***To enter the script that this button will run:*** **1** Click the field, and then click the  button that appears. **2** In the Input Text dialog box, type the JavaScript code in the box. Alternatively, click Browse. In the Open File dialog box, select a file that contains the JavaScript code, and then click Open. Click OK to close the Input Text dialog box.
JavaScript Variable	The JavaScript variable that will be associated with the control. Type any string that qualifies as valid syntax for a JavaScript variable name. The JavaScript variable can be used to exchange a result or value between the protocol and the form. A variable can also be shared between different areas of the same form, for example, to update another field within a form with a button. If you provide a variable, you can specify the Default value for the JavaScript variable.
Default value	Specify the default value of the JavaScript variable.
Always use global context	Specify the context in which to execute the script: • *Global context*. Select the check box to use the global context. For example, if the button is to be pressed while no protocol is running, you must use the global context. **IMPORTANT** Any protocol that uses the global context and contains JavaScript variables of the same name will be affected by changes you make to this variable, even if this form's **Use global context for variables** check box is not selected. • *Context of the running protocol*. Clear the check box to use the protocol's private context.

Property	Description
Minimum user level to interact	See "Properties common to the general controls" on page 212 for a description.
Background color, Font size, Font color	See "Properties common to the general controls" on page 212 for a description.
Experiment tag Step Name	*Experiments database only*. Specifies the name used to create an experiment tag (Step Name) in the Experiments database for the form control when a protocol is run using this form. For details, see "Configuring a form for experiment IDs and Experiments database" on page 249. *Note:* The Experiment tag Step Name property is available only if the Data entry check box is selected.
Experiment tag Field Name	*Experiments database only*. Specifies the name to be used for the corresponding experiment tag in the Experiments database when a protocol is run using this form. For details, see "Adding form controls that allow editing or runtime data display" on page 204. *Note:* The Experiment tag Field Name property is available only if the Data entry check box is selected.
Order in Report	*Experiments database only*. Specifies the order in which the associated data tags for this control appear in an experiments report relative to other controls in the same step. For details on creating forms to track experiment data, see "Configuring a form for experiment IDs and Experiments database" on page 249.

Additional File Browsing control properties

The File Browsing control has the following additional properties.

Property	Description
Button Caption	The text that appears on the control's button. You can use the default (…) or type a custom label.
Show file name including path	The option to show the full file path or only the file name. • Select the check box to display the selected file name including path. • Clear the check box to display only the selected file name.

Property	Description
Assign displayed string to JavaScript variable	Selecting the check box (default) will populate the JavaScript variable with whatever is displayed in the File Browsing control. If the *Show file name including path* check box is selected, then the full path will be displayed. If the *Show file name including path* check box is not selected, then the file name will be displayed without the path. Clearing the check box will populate the JavaScript variable with the full file path regardless of whether the Show file name including path is selected or not. *Note:* This parameter enable protocols made in previous versions of software to get access to the full path in the file browser control, even if *Show file name including path* is unchecked.
Read only, Data entry, Mandatory	See "Properties common to the general controls" on page 212 for a description.

For an example of how to set up a File Browsing control, see "Example: Creating a scripted Pushbutton control in a form" on page 232.

Additional File Browse button properties

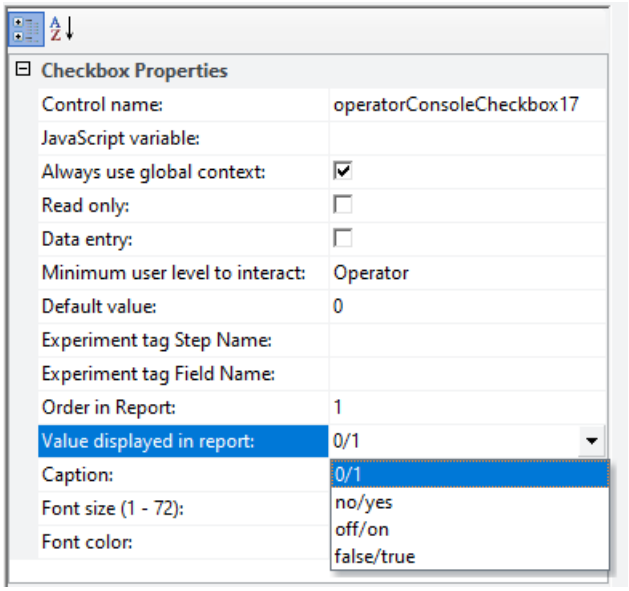
The File Browse button has the following additional properties.

Property	Description
Icon	*Optional*. You may select an image to use as an icon in addition to the button caption. Image files must be imported into the VWorks file storage system before they are available for use in the VWorks software. To use an image in the form: **1** If the image file is not yet in the VWorks file storage, import the image file. See "Importing and exporting miscellaneous files" on page 49. **2** Click the Icon field, and then click the  button that appears. **3** In the Open File dialog box, locate and select the image file (.jpg, .png, .bmp, or .ico), and then click Open.
Caption	To change the label that appears on the button, type the new text in the field.

Property	Description
Enable when paused	• Select the check box if you want the button to be enabled while a running protocol is associated with the form. This parameter defaults to true (check box selected) when the control is created, or when opening a form in the Form Editor that was created in a version earlier than VWorks 14.1.1. • Clear the check box to disable the button while a running protocol is associated with the form, even if the protocol is paused.

Checkbox properties

The Checkbox control has the following properties in addition to the properties common to the general controls.



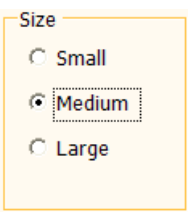
Property	Description
Caption	Type the string to be displayed to the right of the check box.
Default value	Type one of the following: 0 — indicates the check box is cleared by default 1 — indicates the check box is selected by default *Note*: Any value other than zero indicates that the check box is selected.

Property	Description
Value displayed in report	*Experiment ID reports only*. The value that appears in the experiment ID report when the corresponding check box on the form is cleared or selected for a protocol run. For details on setting up a form to work with the Experiments database, see "Configuring a form for experiment IDs and Experiments database" on page 249. Select one of the following options: • 0/1 • no/yes • off/on • false/true

The Checkbox control is binary. When the check box on the form is selected, a 1 is written to the associated JavaScript variable. When the check box is cleared (unchecked), a 0 is written to the associated JavaScript variable.

Radio Button properties

The Radio Button control has the following properties in addition to the properties common to the other general controls.

Property	Description
Default value	Type one of the following: 0 — indicates the option is cleared by default 1 — indicates the option is selected by default *Note:* Any value other than zero indicates that the check box is selected.
Caption	Type the string to be displayed to the right of the radio button.
Group ID	Type a string to use as the ID of the radio button group. Only one button in the group can be selected at a time. When the form's user selects a button in the group, all other radio buttons with the same Group ID are cleared (not selected) automatically. For example, if you had a grouping of options for size, you could use the string, `size`, as the Group ID.  *Note:* You can use a Group box to visually group a collection of radio button options.

The Radio Button control is binary. When the button on the form is selected, a 1 is written to the associated JavaScript variable. When the button is cleared, a 0 is written to the associated JavaScript variable.

Note: If the JavaScript variable's numeric value is anything other than 0, the option will display as selected.

Image (static) control properties

The Image control has the following properties in addition to the properties common to the other general controls.

Property	Description
Image file	Image files must be imported into the VWorks file storage system before they are available for use in the VWorks software. To import an image file, see "Importing and exporting miscellaneous files" on page 49. To use the image in the form, click the Image file field, and then click the  button that appears. In the Open File dialog box, locate and select the image file (.jpg, .png, .bmp, or .ico), and then click Open.

Date and Time properties

The Date and Time control has the properties common to the other general controls. However, if you provide a default value, it must be in the following format. If you do not provide a default value, the software uses the current date and time from the computer.

Property	Description
Default value	nn:nn:nn nn-nn-nnnn where, nn:nn:nn is the hour:minutes:seconds in a 24-hour clock and nn-nn-nnnn is the month:day:year

If you select the Data entry property for this control, the user can enter:

- *Time*. The user can edit the default values.
- *Date*. The user can click the drop-down list to select the date from a calendar.

Group Box properties

The Group Box control has the following properties in addition to the properties common to the other general controls.

Property	Description
Group name	Type the string to display as the caption for the group box.

Controls that you drag inside the group box become part of the group box. The group box may need to be resized to ensure that all the controls within the group box borders are clearly visible.

Tab Control properties

The Tab Control has the following properties in addition to the properties common to the other general controls.

Tab Control property	Description
Alignment	Specifies the position of the tabs: Top, Left, Right, or Bottom.
SizeMode	Specifies the tab size: • *Normal*. Each tab is sized to fit its name. • *Fixed*. The longest tab name determines the size of all the tabs. • *FillToRight*. The longest tab name determines the size of all the tabs, and the tabs are flush with the right and left margins of the control.
Width	Specifies the width of the tab control.
Height	Specifies the height of the tab control.

IMPORTANT

The Dock selection overrides the values you specify for the Width and Height properties. For example, if you dock the control at the top of the canvas, the software increases the width of the control and updates the Width value. To return to the previously set Width and Height values, set the Dock property to None. For more information about docking tabbed pages, see "Properties common to the general controls" on page 212.

Each tabbed page contains the following properties in addition to the properties common to the other general controls.

Tabbed page property	Description
Name	The text that is displayed on the tab. Provide a name that describes the function of the tab contents.

Panel properties

The Panel control has properties that are common to the other general controls. See "Properties common to the general controls" on page 212.

Related information

For information about...	See...
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
Pushbutton and File Browsing controls	"Example: Creating a scripted Pushbutton control in a form" on page 232
Controls under Specialized Buttons in the Form Designer window	"Overview of the specialized buttons" on page 188
Controls under Specialized Display Controls in the Form Designer window	"Adding indicators for elapsed time and progress to a form" on page 202
Form properties in the Form Designer window	"Setting the form properties" on page 240
JavaScript context and form design	"Understanding JavaScript context in form design" on page 243
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Example: Creating a scripted Pushbutton control in a form

To create or edit forms, you must be an administrator or technician. You should also have an understanding of how to create protocols in the VWorks software.

This topic provides an example of the Pushbutton and File Browsing controls in the Form Designer. For more details, see "Adding form controls that allow editing or runtime data display" on page 204.

About the script for the example

The example presented in this topic uses the following script:

```
runset.appendProtocolFileToRunset (file01, numRunTimes, "",
"myform1.VWForm");
```

This script appends the contents of a protocol file to the Runset Manager, specifies the number of times the protocol should run, and passes the file name of the form (.VWForm) to be associated with the protocol.

The following table describes the arguments for the `appendProtocolFileToRunset` method used in the example script.

Argument	Example	Descriptions
string *ProtocolPath*	`file01`	The protocol file path. In this example, the variable `file01` is used for the file path.
int *RunTimes*	`numRunTimes`	The number of times (integer) to run the protocol. In this example, we use the variable `numRunTimes` so that the form users can edit this value.
string *ProtocolNotes*	`""`	An empty string. Although the third argument is not used in this case, the empty string is required as a placeholder.
string *formToUse*	`"myform1.VWForm"`	The file name of the form to be associated with the protocol once the protocol starts running. The argument must specify the file name (not the path) of the form that contains the pushbutton.

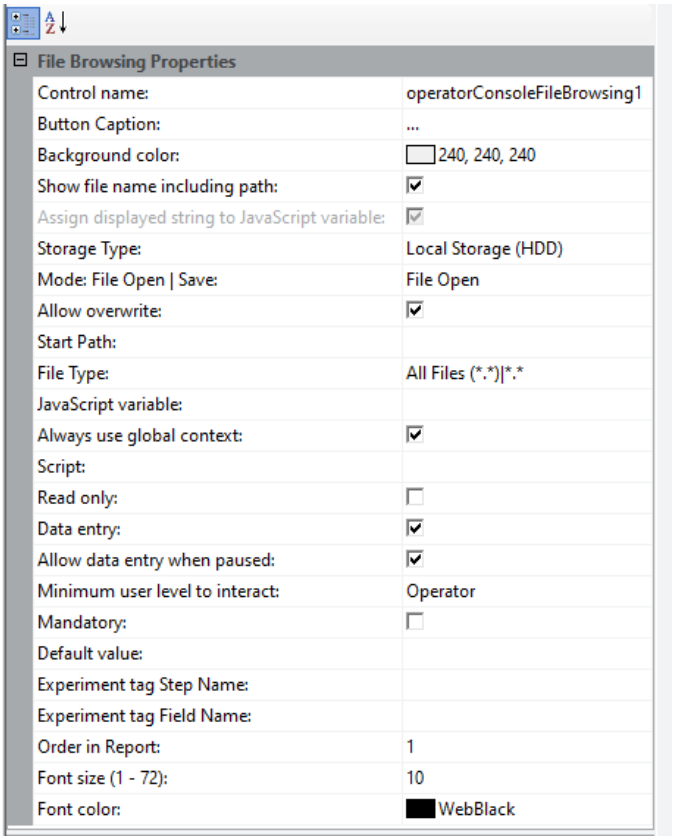
For details on other runset object methods that can be used with forms, see "runset object" on page 148.

Creating the example pushbutton

The following example describes how to create a pushbutton that enables the form's user to select which protocol to run.

To create the example pushbutton:

1. In the **VWorks** window, choose **File > New > Form**.
2. In the **Form Designer** window, click **Save As** and save the form with the name `myform1.VWForm`.
3. To configure the File Browsing control, drag **File Browsing** from **General Controls** to the **Form Canvas**. Enter the following settings under **File Browsing Properties:**



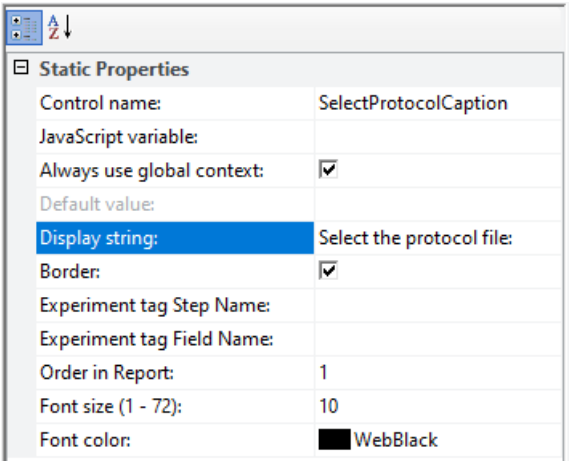
Property	Value
Storage Type	Select the storage type for your VWorks edition: • Shared Service Repository — Secure Storage • Local Storage (HDD) — local file storage
Mode File Open|Save	Select File Open.
JavaScript variable	Enter `file01`.
Start Path	Enter the file path for the file that this control will open, for example, [olssvr]:VWorks Projects/VWorks/Protocols.
Always use global context	Select the check box.
Data entry	Select the check box.

Use the default selections for the remaining properties.

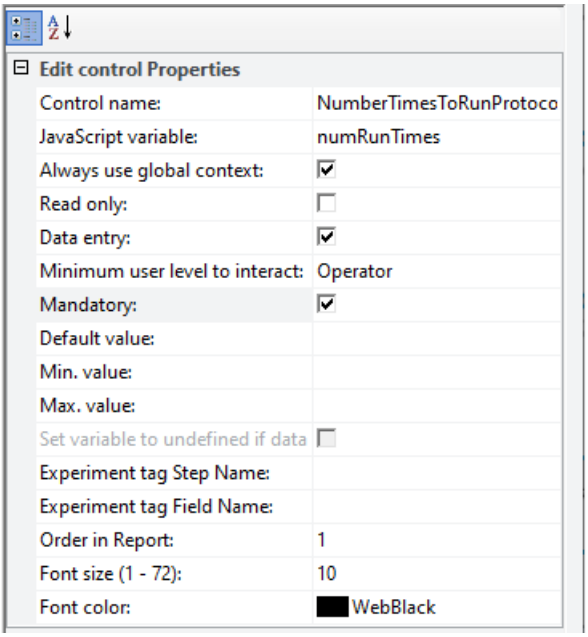
4 *Optional*. To provide a caption for the File Browsing control, drag **Static** from **General Controls** to the **Form Canvas**.

a Under **Static Properties**, in the **Control Name** field, type a functionally descriptive name for the control.

b In the **Display string** field, type the following caption: `Select the protocol file:`



5 To configure an Edit control that allows users to enter the number of runs, drag **Edit control** from **General Controls** to the **Form Canvas**. Enter the following settings under **Edit control Properties:**



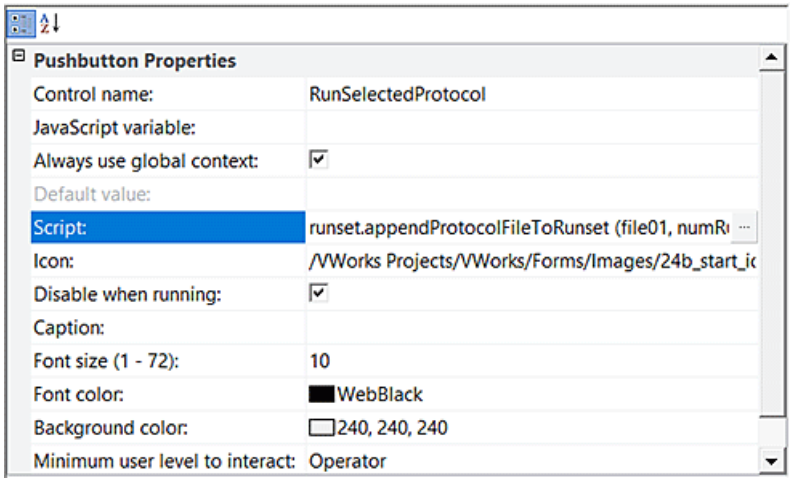
Property	Value
JavaScript variable	`numRunTimes`
Always use global context	Select the check box.
Data entry	Select the check box.
Mandatory	Select the check box.

Note: The Mandatory setting causes the software to display an error message If the form's user forgets to enter the number of runs.

You can also enter a **Default value** that the software will use unless the form user changes the value.

6 *Optional*. Repeat step 4 to create the following caption for the number-of-runs control: `Number of times to run:`

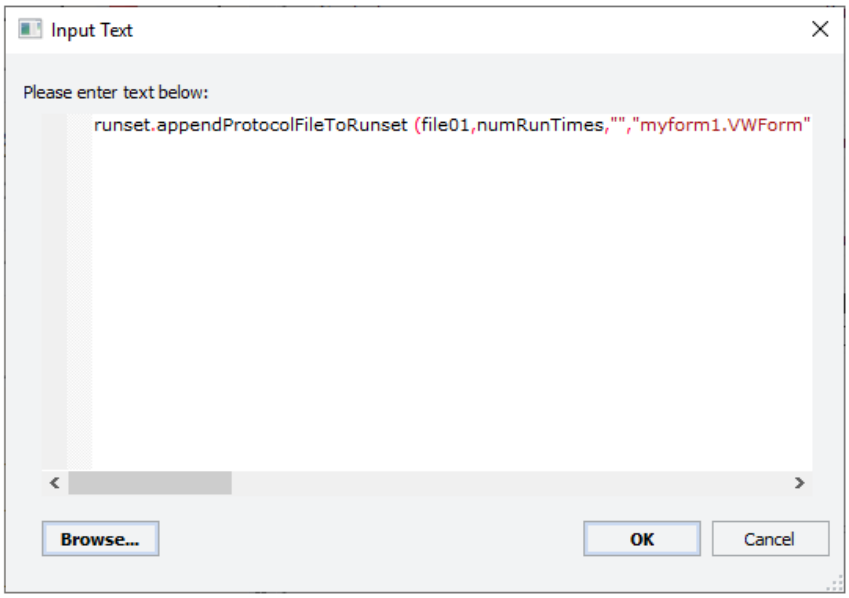
7 To configure the pushbutton, drag **Pushbutton** from **General Controls** to the **Form Canvas**. Enter the following settings under **Pushbutton Properties**:



a Click the **Script** field, and then click the ... button that appears. In the **Input Text** dialog box, type the following script:

```
runset.appendProtocolFileToRunset (file01, numRunTimes,
"", "myform1.VWForm");
```

Note: If you copy this example, make sure to use straight quotation marks instead of curly quotation marks or smart quotes.

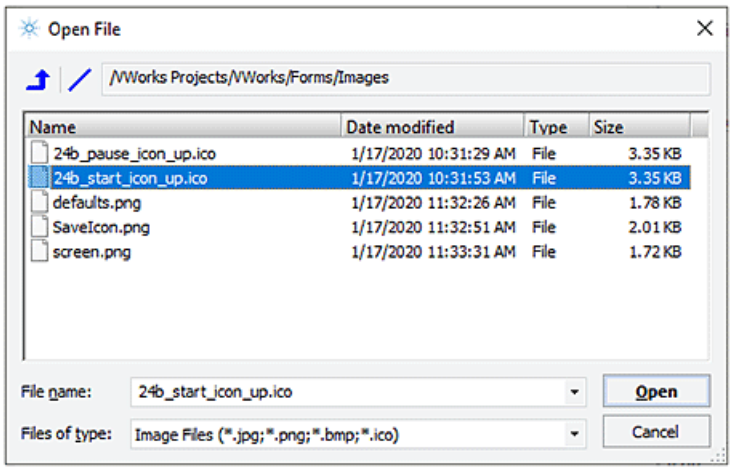


b Select **Always use global context** and **Disable when running**.

IMPORTANT The global context is required for the variables in the Pushbutton script, the File Browsing control, and the Edit control because at the time the button is pressed the form is not yet associated with a protocol. Without a protocol association, there is no protocol from which to get an associated protocol's context.

c Type the **Caption** that will appear on the pushbutton, for example, Run Selected Protocol.

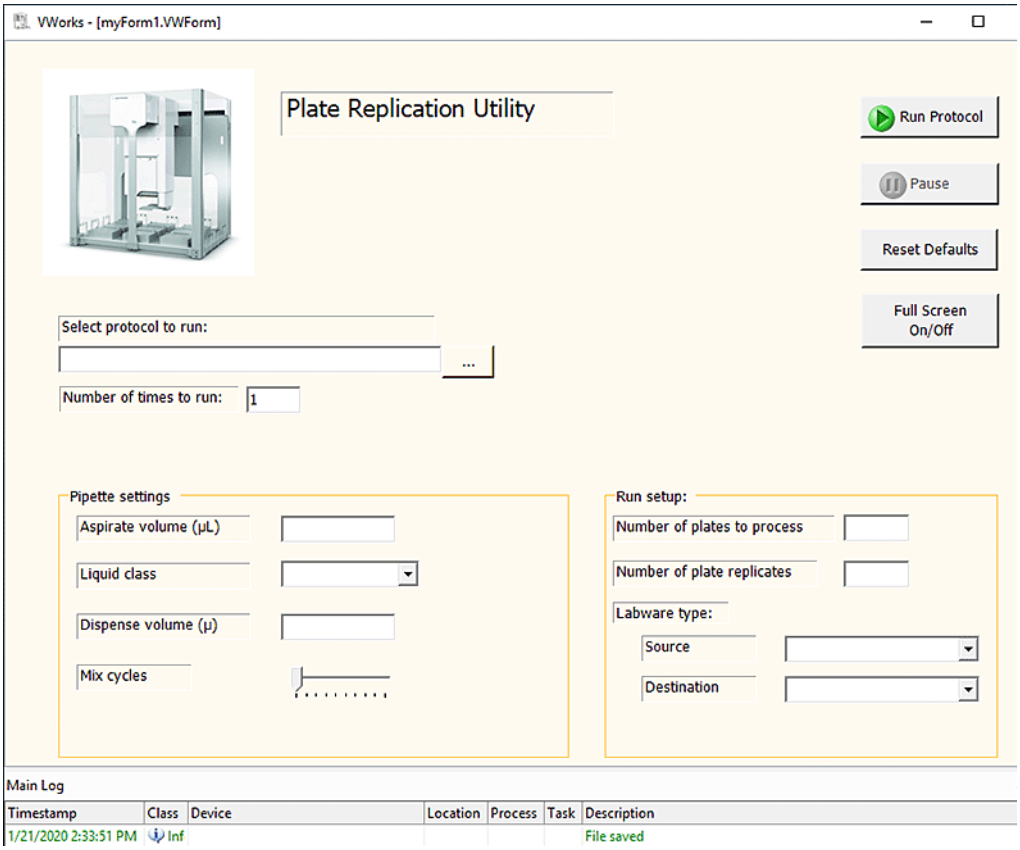
d To select an icon for the Pushbutton control, click the **Image** field, and then click the ... button that appears. In the **Open File** dialog box, select the following icon file (.*ico) for the run button. This example uses 24b_start_icon_up.ico.



Note: Image files must be imported into the VWorks file storage system before they are available for use in the VWorks software. To import an image file, see "Importing and exporting miscellaneous files" on page 49.

8 Click **OK** to close the Form Designer and save the changes.

Figure Example form in full screen view with the logs included



Testing the example pushbutton

To test the example pushbutton:

1. Ensure *myform1.VWForm* is open in the **VWorks** window.
2. In the form, click ... and then select the protocol file (.pro).
3. In the **Number of times to run** box, type a value.
4. Click the custom pushbutton that you created, for example, **Run Selected Protocol**.

Related information

For information about...	See...
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
Controls under General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Controls under Specialized Buttons in the Form Designer window	"Overview of the specialized buttons" on page 188

For information about...	See...
Controls under Specialized Display Controls in the Form Designer window	"Adding indicators for elapsed time and progress to a form" on page 202
Form properties in the Form Designer window	"Setting the form properties" on page 240
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using JavaScript	"Creating a protocol: advanced topics" on page 89

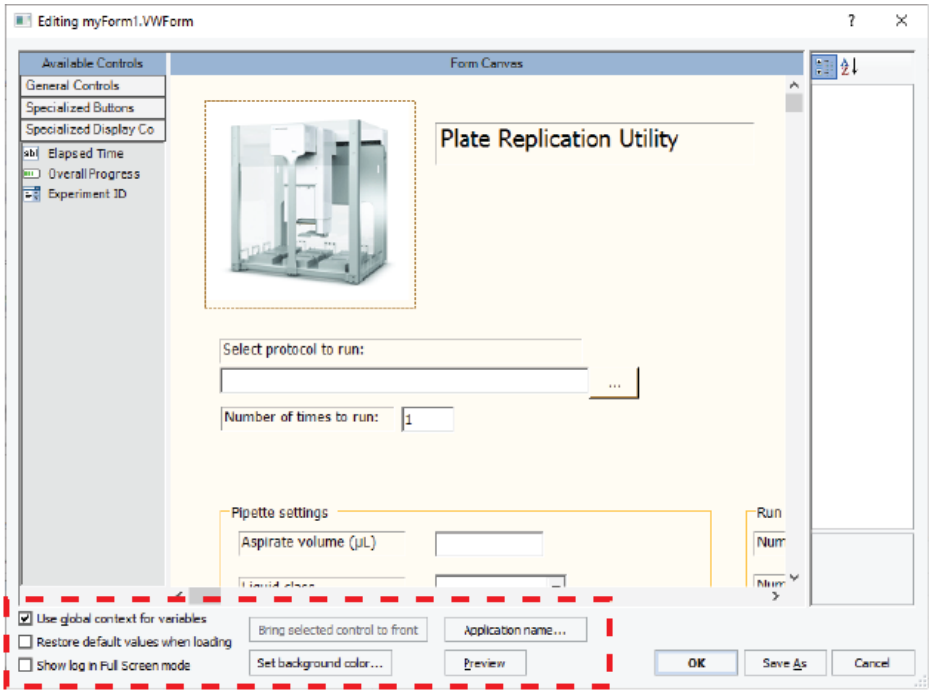
Setting the form properties

About this topic

To create or edit forms, you must be an administrator or technician. You should also have an understanding of how to create protocols in the VWorks software.

This topic describes the form properties that appear at the bottom of the Form Designer windows.

Figure Form properties in the Form Designer and Editing .VWForm windows



For more details on creating forms, see "Workflow for creating or editing a form" on page 181.

Procedure

To set the form properties:

1 In the **Form Designer** window or **Editing .VWForm** window, specify the context for the JavaScript variables associated with all the controls in the form:

- *Context of the running protocol*. Clear the **Use global context for variables** check box. JavaScript variables with the same name in different protocols will not be affected by the properties you set for the controls in this form.

  Individual controls in the form can still specify the global context.

- *Global context for all variables in the form*. Select the **Use global context for variables** check box. For example, if the form is for a runset, you must use the global context.

  All controls in the form will use the global context for their JavaScript variables and scripts, regardless of the settings for the individual control properties.

Any protocol that uses the global context and contains JavaScript variables of the same name will be affected by the changes you make to the properties for the controls in this form.

CAUTION **Ensure that the variable values you want to use globally are applicable to all other protocols, which also specify the global context. Otherwise, variables with the same name in different protocols will interfere with each other.**

For more details on JavaScript context, see "Understanding JavaScript context in form design" on page 243.

2 To ensure that the default values associated with the controls in the form appear each time a user opens the form, select **Restore default values when loading**.

3 If the form includes the Toggle Full Screen button, specify whether to show or hide the Main Log, Pipette Log, Time Constraints Log, and Progress tabs in the full screen view:

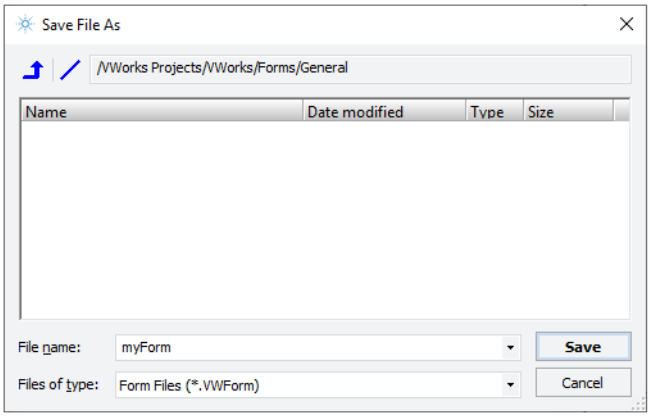
- Select the **Show log in Full Screen mode** check box to include the logs.
- Clear the **Show log in Full Screen mode** check box to hide the logs.

4 To change the background color of the form, click **Set background color**, and then select the color from the **Color** dialog box.

5 If you are tracking experiment data, click **Application name** to specify the application. For details, see "Workflow to configure a form to track experiment IDs" on page 249.

6 Click **OK**.

- If this is the first time to save the form, the Save File As dialog box opens. Type a name for the form, and click **Save**.



VWorks Plus only. If an audit trail is being logged, select or type a reason for change in the **Audit Comment** dialog box, and then click **OK**.

- If you are editing a previously saved form, the software saves the form and closes the Form Designer.

  VWorks Plus only. If an audit trail is being logged, select or type a reason for change in the **Audit Comment** dialog box, and then click **OK**.

Related information

For information about...	See...
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Specialized Buttons in the Form Designer window	"Overview of the specialized buttons" on page 188
Specialized Display Controls in the Form Designer window	"Adding indicators for elapsed time and progress to a form" on page 202
JavaScript context and form design	"Understanding JavaScript context in form design" on page 243
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Understanding JavaScript context in form design

This topic assumes a basic understanding of how to use JavaScript variables and create protocols in the VWorks software.

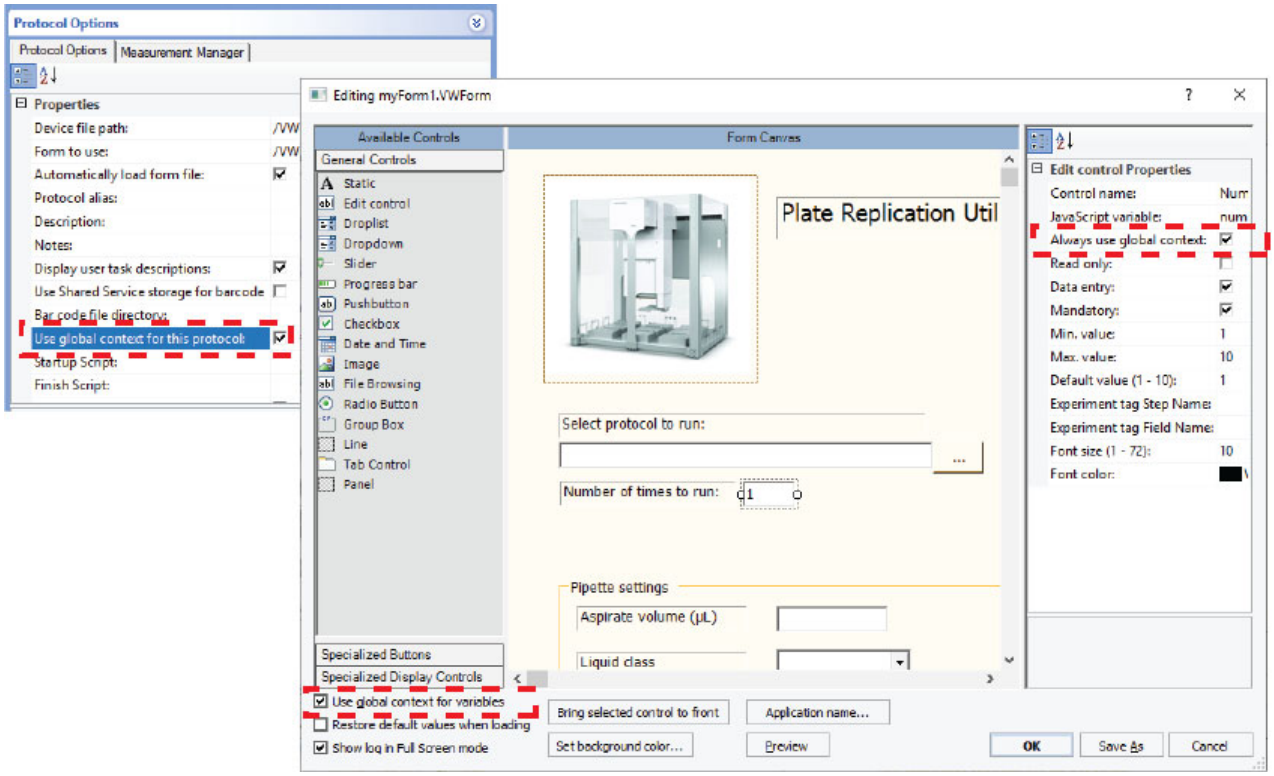
About JavaScript context and form design

The JavaScript variables used in protocols and in the forms that run the protocols can be in either the global context (available for all protocols that use global context for variables) or limited to the context of the running protocol.

When designing a form, you specify the JavaScript context in three locations:

1. *Protocol*. In the **Protocol Options** area, select or clear the **Use global context for this protocol** check box to specify the context for the variables in the protocol.
2. *Form*. In the Form Designer window, select or clear the **Use global context for variables** check box to specify the context for the variables in the form.
3. *Individual controls in the form*. If the form **Use global context for variables** check box is selected, the context setting for all the controls that use JavaScript is automatically set.

   If the form **Use global context for variables** check box is not selected, you can select the **Always use global context** check box in the control properties for individual controls that use JavaScript.



Protocol JavaScript context

For each protocol, you specify the context in the Protocol Options area:

- *Global context*. Variables of the same name in the given protocol will be available across all protocols that also specify the global context.

- *Context of the running protocol*. Prevents the variables in a given protocol from being impacted by changes to variables of the same name in other protocols.

JavaScript context of form

In the Form Designer window, you select or clear the **Use global context for variables** check box to specify the context for the form.

- *Global context*. All variables throughout the entire form will be in the global context.
- *Context of the running protocol*. The variables in each individual control in the form will be in the context of the associated (running) protocol, unless the control specifies the global context. When no protocol association exists (the protocol is not running or scheduled to run), the individual controls that do not specify global context will retain their most recent previous value.

  The global context may not be necessary if the variable in question will communicate with the protocol after the protocol is running or is scheduled to run.

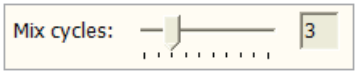
No JavaScript objects that exist in a context other than the global context and the protocol's context will be accessible from either kind of form. For example, plate objects, plates[] array objects, and task objects will not be accessible. If you want the data from any of these objects to be shown on the form, the protocol must contain JavaScript statements at appropriate points to copy the relevant data into JavaScript variables that you devise.

Individual control JavaScript context

In the Form Designer window, when you set the properties for a control that has an assigned JavaScript variable, you select or clear the **Always use global context** check box to specify the context.

- *Global context*. The variable in question will be available to the protocol and to other controls that also use the global context before the protocol is running or is scheduled to run.

  If two controls on the form must communicate with each other, the global context is required for each control. For example, the following figure shows a Slider control and a display (Static text control). As the slider moves, the value in the display updates to match the slider's position.



- *Context of the running protocol*. The variable in question will be available to the protocol only after the protocol is running or is scheduled to run.

Runsets and the global context

If you have a set of protocols designed to be a runset that will be launched by a form, you use the global context for both the protocols and the form.

Note: Alternatively, you can use the JavaScript function, GetGlobalObject, to access the global context for any variable within the protocol that should interact with the form. See "VWorks-defined functions" on page 103.

IMPORTANT

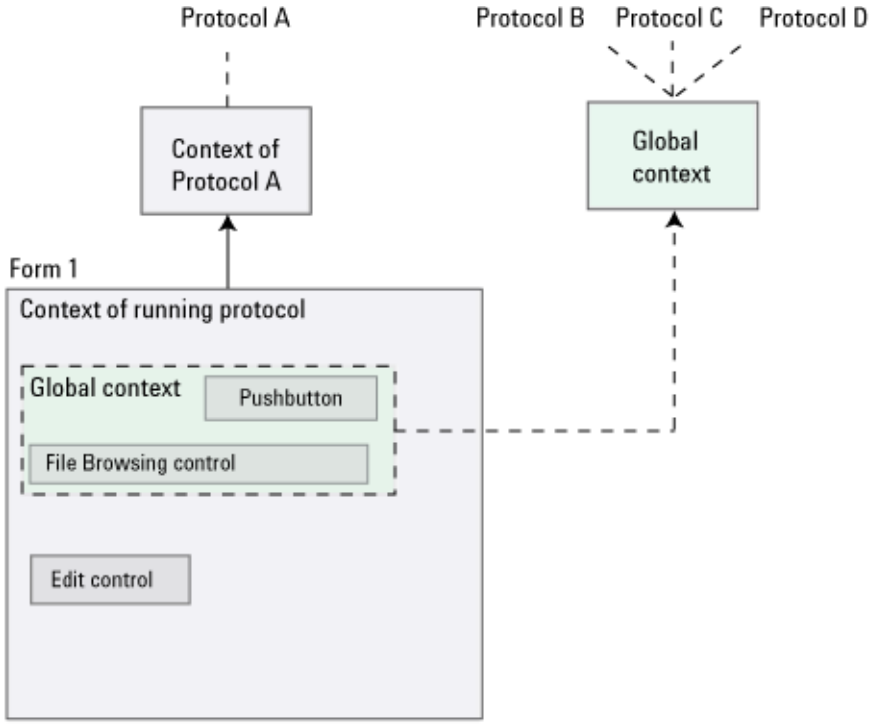
Multiple protocols designed to run together in the same runset should not reuse the same JavaScript variable names unless the variables are intended to share data.

Context scenarios

The following figure shows four protocols (A, B, C, and D). Form 1 runs protocol A, and protocols B, C, and D run without using a form.

- Protocol A uses its own context for variables.
- Protocols B, C, and D use the global context for variables.
- Form 1 uses the context of the running protocol, except for two of its controls, a pushbutton and a file browsing control, which use the global context.

Figure Example of relationships between the protocol, form, and control context settings



Suppose that protocol A does not use the global context because it uses a variable `x` to count the number of times a loop has executed. But, protocol D also uses a variable called `x` for something completely different. Even though both protocols may run at the same time, their JavaScript variables will not interfere with each other because protocol A uses its own private context.

Suppose that protocols B, C, and D must use the global context because each of these protocols keeps track of the total volume dispensed by all protocols in a variable called `totVol`. Whenever any of these protocols executes a Dispense task, it also executes the JavaScript: `"totVol += task.volume;"`. When all three protocols are done, the total volume dispensed in all three protocols can be accessed in a single variable.

In Form 1, the File Browsing control and Pushbutton control are in the global context so that so that the two controls can communicate with each other. The Pushbutton control executes a script that uses a JavaScript variable of `fileName`, which must also be specified by the File Browsing control. Now imagine that the user has typed, `C:\myfile.txt` in this control. When the user presses the Pushbutton control, whatever string happens to be showing in its File Browsing control at that time is assigned to the variable `fileName` in the global context. Now, not only does the script in the Form 1 Pushbutton control recognize the variable `fileName` as having a value of `C:\myfile.txt`, so would all the scripts throughout protocols B, C, and D.

Related information

For information about...	See...
Creating or editing a form	“Workflow for creating or editing a form” on page 181
Controls under General Controls in the Form Designer window	“Adding form controls that allow editing or runtime data display” on page 204
Form properties in the Form Designer window	“Setting the form properties” on page 240
Creating a basic protocol	“Workflow for creating a basic protocol” on page 22
Working with JavaScript	“Creating a protocol: advanced topics” on page 89

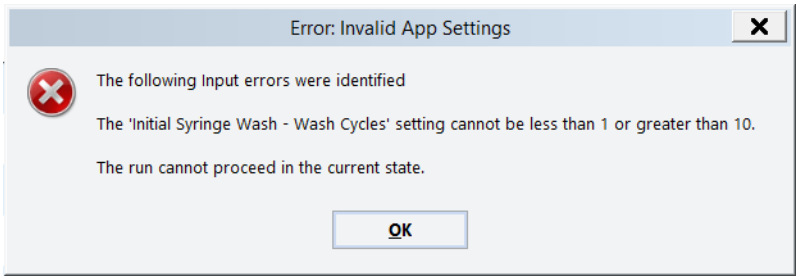
Opening a message from a form button using JavaScript

This topic describes the VWorks-defined JavaScript function, `DisplayAsyncMessageBox()`, which can be called from a script that is run from a button on a form. For example, this function could be called from a script in the Run Specified Script button or the Pushbutton control on a form before running the specified protocol.

IMPORTANT Do not call DisplayAsynMessageBox() from a script within a protocol. Doing so could cause the protocol to fail or have unexpected results.

DisplayAsyncMessageBox()

Calling `DisplayAsyncMessageBox()` displays a message dialog box with an OK button, as the following example shows.



The call returns immediately and does not wait for the user to press the OK button.

Parameters

- *Text string*. Required. The text to display in the message dialog box.
- *Text string*. Required. The caption at the top of the message dialog box.
- *String*. Optional. A string specifying the icon to be displayed in the message dialog box.

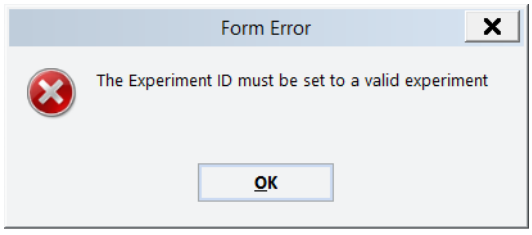
  The only options for this third string are:

String option	Icon displayed
"error"	Error icon (white X on red circle)
"warn" or "warning"	Warning icon (black ! on yellow triangle)
"info" or "information"	Information icon (white i on blue circle

Examples

```
DisplayAsyncMessageBox("The Experiment ID must be set to a valid experiment.","Form Error","error")
```

The following figure shows the resulting message dialog box.




```
DisplayAsyncMessageBox("To prevent potential injury, keep
clear of the device while it is in
motion.","Warning","Warning")
```

```
DisplayAsyncMessageBox("Please wait. The run will begin in a
few seconds.","Form Information","")
```

Related information

For information about...	See...
Creating or editing a form	"Workflow for creating or editing a form" on page 181
Controls under General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Form properties in the Form Designer window	"Setting the form properties" on page 240
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Working with JavaScript	"Using JavaScript" on page 100

Configuring a form for experiment IDs and Experiments database

Before you start

To create or edit forms, you must have VWorks technician- or administrator-level access. You should also have an understanding of how to create protocols in the VWorks software.

This topic assumes that you have already established a connection to the Experiments database. For instructions on how to set up and connect to the Experiments database, see the *VWorks Automation Control Setup Guide*.

Workflow to configure a form to track experiment IDs

Item	For this task...	See...
1	Create the form. *Note:* If you want the form variables to be in the global context, ensure that the **Use global context** check box is selected for the form.	"Workflow for creating or editing a form" on page 181
2	In the Form Designer window or Edit *<form name>*.VWForm window, specify the Application name that will be associated with the experiment ID on this form.	"Specifying the application name for the form" on page 250
3	Add the desired buttons and controls for the Experiments Editor: • Select Experiment ID button and Experiment ID control • Menu Action buttons, such as Experiments Editor, Add Experiment Note, and Experiments Archive • Generate Experiment ID Report button • Experiment ID droplist *Note:* If a form has the Experiment ID control, it should not also have an Experiment ID droplist, and vice-versa.	"Adding specialized controls for the Experiments Editor" on page 253
4	Specify the following properties for any form controls to be tracked in the Experiments database: • Experiment tag Step Name • Experiment tag Field Name • Order in Report *Note:* Typically, these are the controls that also have the Data entry property.	"Editing the Experiment tag properties in form controls" on page 251
5	Edit the protocol to include a Step Started/Completed task before and after each Step (task sequence) to be tracked.	"Configuring a protocol to track experiment data" on page 260

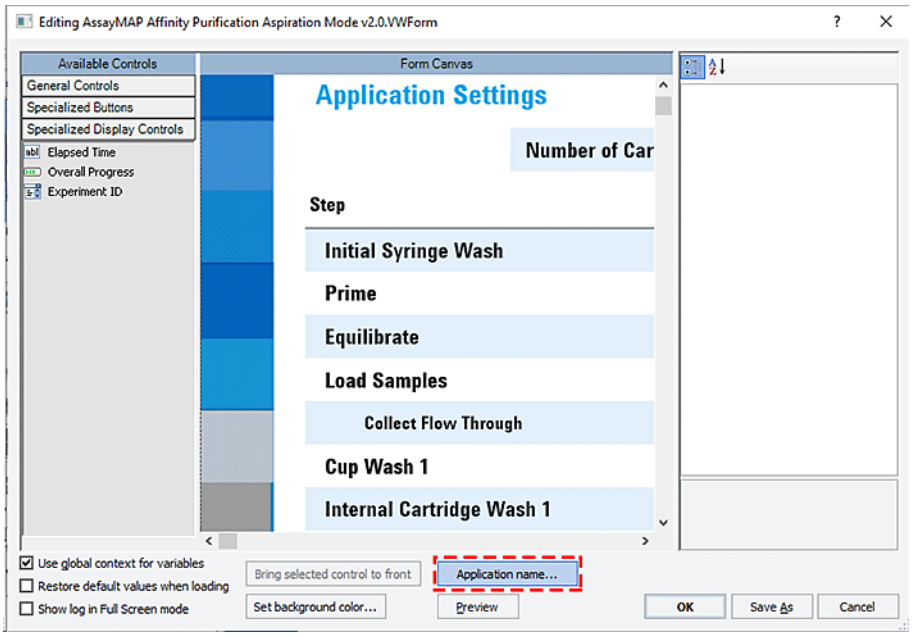
Specifying the application name for the form

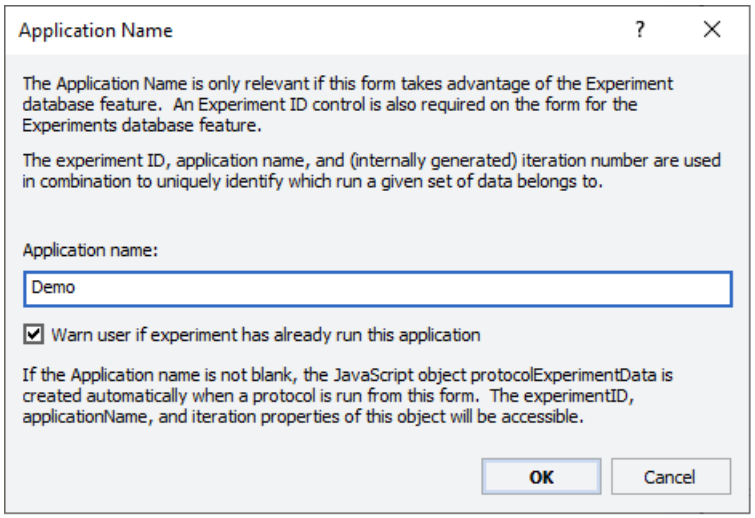
IMPORTANT

The following settings take effect only if the form also contains an Experiment ID control.

To specify the application name associated with the form:

1 In the **Form Designer** window or **Editing *<form name>*.VWForm** window, click **Application name**. The Application Name dialog box opens.





2 In the **Application Name** dialog box, specify the following:

- **Application name**. The name of the application that this form governs.

  Type a name that does not start with an = sign or contain any single-quote (') or double-quote (") characters.

When a protocol is run using this form, the software uses this application name for all the associated experiment tags added to the Experiments database.

- **Warn user if experiment has already run this application**
  - Select this check box to display a warning message if the form's user clicks the run button, but an iteration has already been run for this experiment ID and application name combination.
  - Clear this check box if you do not want to display a warning message when an iteration has already been run for this experiment ID and application name combination.

Note: When a VWorks form that has both an Experiment ID control and an Application name runs a protocol, the protocolExperimentData object is created automatically. The protocolExperimentData object has five properties: experimentID, applicationName (string), iteration (int), lastStartedStepName (string), and stepCompleted (boolean). The values of these properties are populated at run time. This object is not created automatically if the same protocol is run without the form.

3 Click **OK**.

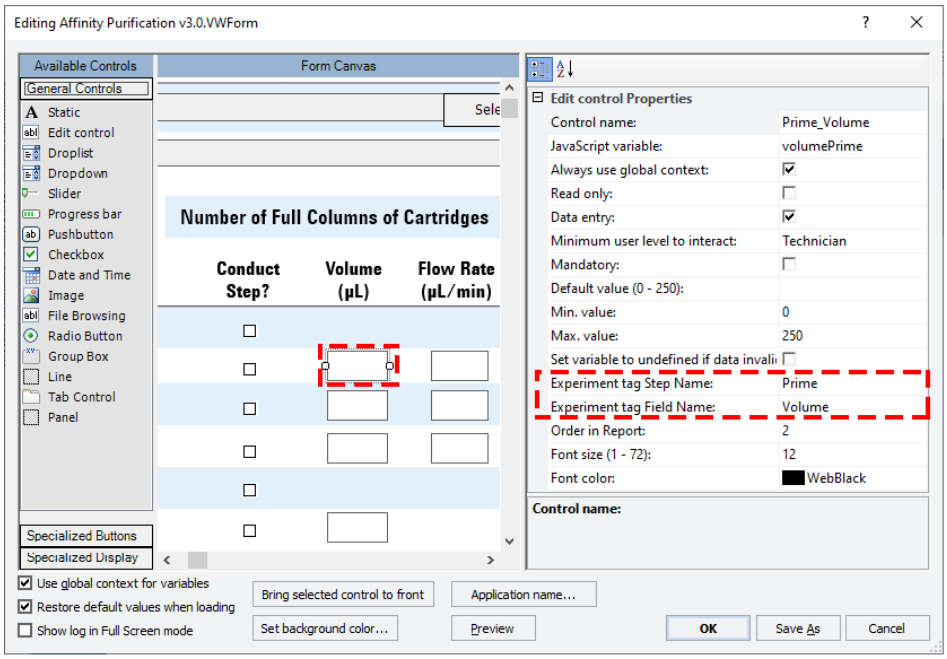
Editing the Experiment tag properties in form controls

All form control types that have a Data Entry property have the following Experiment tag properties.

Property	Description
Experiment tag Step Name	The step name used to create an experiment tag (Step Name) in the Experiments database for this form control when a protocol is run using this form. Enter a name that matches the corresponding task sequence (bracketed by the Step Started/Completed tasks) in the protocol. For example, the step name of a Bravo Subprocess could be Equilibrate, Load Sample, Wash, and so forth. If this Step Name property is blank, the Step Name tag for this control will be blank in the Experiments database. The blank property means that the experiment tag in question is not associated with any given step in the running protocol. The following figure shows a step name example of Prime.
Experiment tag Field Name	The field name used to create an experiment tag (Field Name) in the Experiments database for this form control when a protocol is run using this form. Enter a name that matches the corresponding task property within the tagged task sequence (Step) in the protocol. For example, if the Step Name field is Equilibrate, the Field Name could be Volume or Flow Rate. If the Step Name field is blank, the value for the Field Name property might be Number of Runs, Protocol To Run, number or pipette tips, and so forth. *Note:* If the Field Name property is blank for the given control, the experiment tag is not created for this control. The following figure shows a field name example of Volume.

Property	Description
Order in Report	The order in which the associated data tags for this control appear in an experiments report relative to other controls in the same step, where 1 = first, 2 = second, 3 = third, and so forth The following example shows the Order in Report setting is 2 for the Prime Volume step.

Figure Example of Experiment tag and Order in Report properties



Related information

For information about...	See...
Creating or editing a form	"Workflow for creating or editing a form" on page 181
Controls under General Controls in the Form Designer window	"Adding form controls that allow editing or runtime data display" on page 204
Form properties in the Form Designer window	"Setting the form properties" on page 240
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Working with JavaScript	"Creating a protocol: advanced topics" on page 89

Adding specialized controls for the Experiments Editor

The Experiments Editor controls include:

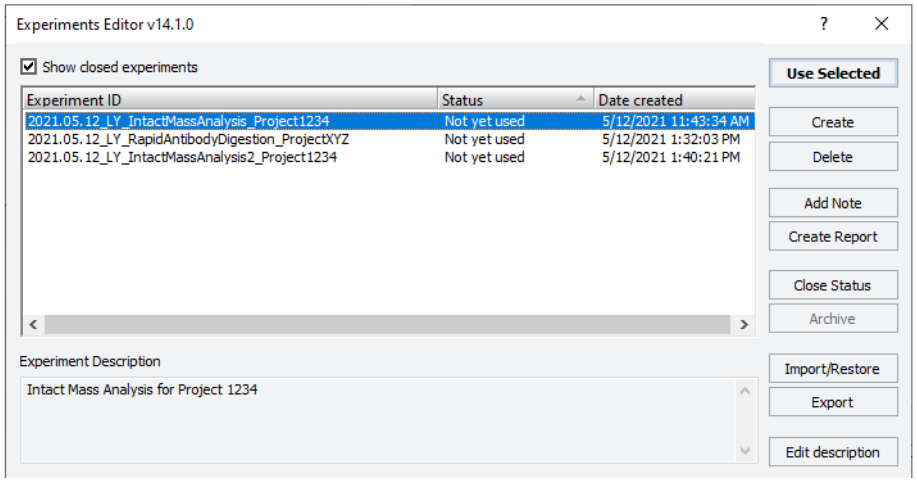
- Select Experiment ID button and Experiment ID control
- Menu Action buttons to open Experiments Editor, Add Note dialog box, and Experiments Archive dialog box
- Generate Experiment ID Report button
- Experiment ID droplist

Select Experiment ID button and Experiment ID control

You can pair the following controls to select an experiment and display the selected experiment ID in the form:

- **Select Experiment ID button**. Opens the Experiments Editor in the Choose Experiment mode. In this mode, the Experiments Editor has an additional Use Selected button.

Figure Experiments Editor in Choose Experiment mode



When the form's user selects the Experiment ID using the Select Experiment ID button, the ID is populated to any Experiment ID control on the form.

- **Experiment ID control**. Displays the experiment ID that is selected using the Select Experiment ID button.

Note: Alternatively, you can include the Experiment ID droplist on the form, which allows the form's user to select an existing experiment ID or open the Experiments Editor and create an experiment ID. If a form has an Experiment ID control, it should not also have an Experiment ID droplist, and vice-versa.

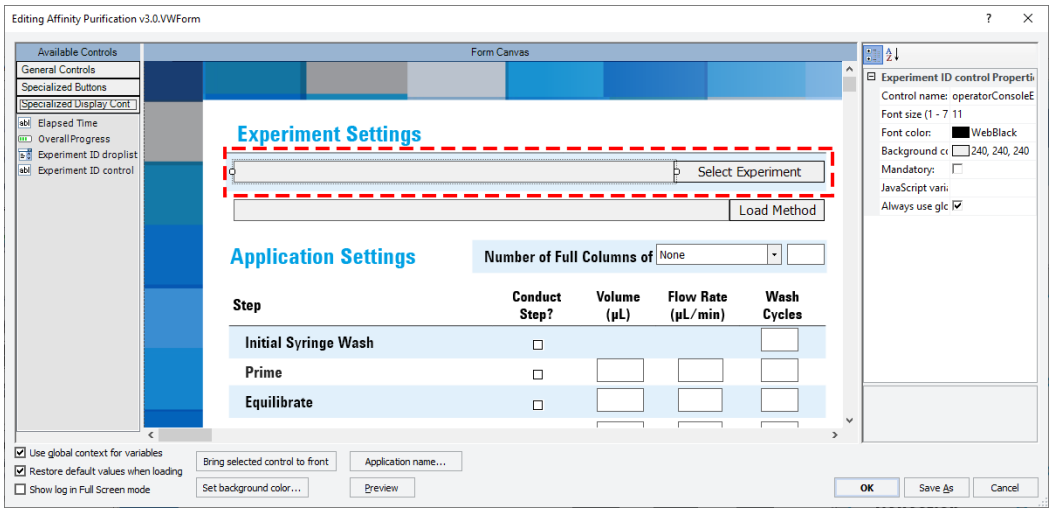
To add the Select Experiment ID button to the form:

1. In the **Form Designer** window or **Editing*<form name>*.VWForm** window, under **Available Controls**, click **Specialized Buttons**, and then drag the **Select Experiment ID** button to the form canvas area.

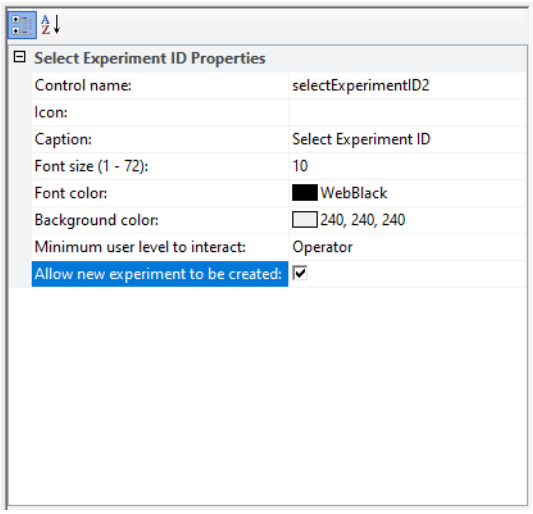
   Note: To turn off the autogrid, press ALT while dragging the control into position.

2. In the **Form Canvas** area, click the control so that a resizing box appears around the border. To resize the control, drag one of the sizing handles.

Figure Example of Experiment ID control (left) and Select Experiment ID button (right)



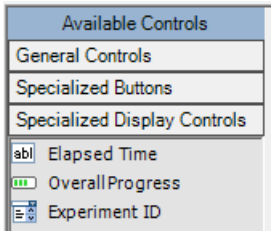
3 Specify the property values:



Property	Description
Allow new experiment to be created	Selected by default. • If check box is selected — When the form's user clicks this button control, the Experiment Editor opens with both the Create and Import/Restore buttons enabled. • If check box is cleared — When the form's user clicks this button control, the Experiment Editor opens with both the Create and Import/Restore buttons disabled, preventing the user from creating a new experiment ID.
All other properties	See "Properties specialized buttons have in common" on page 195.

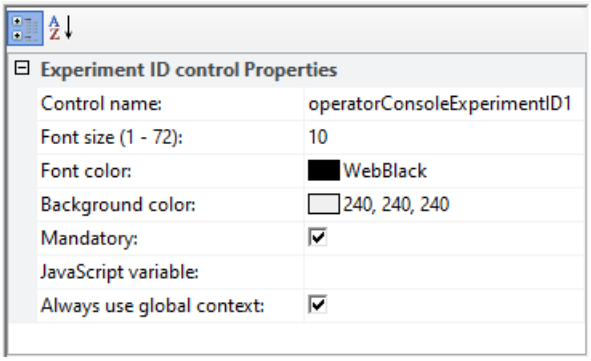
To add the Experiment ID control to the form:

1 In the **Form Designer** window or **Editing<*form name*>.VWForm** window, under **Available Controls**, click **Specialized Display Controls**, and then drag the **Experiment ID control** to the form canvas area.



2 In the **Form Canvas** area, click the control so that a resizing box appears around the border. To resize the control, drag one of the sizing handles.

3 Specify the following properties:



Property	Description
Control	The descriptive name associated with the control. The Control name does not appear on the form. Use a functionally descriptive name so that references to this control are recognizable on audit trail reports. For details, see "About the naming of form controls" on page 184.
Font size	The size of the text that this control will display after the form's operator selects the experiment ID.
Font color	The color of the text that this control will display after the form's operator selects the experiment ID. Click the box to select the color.
Background color	The background color or the control. Click the box to select the color.

Property	Description
Mandatory	The option to require the form's user to enter a value for this field before starting the associated protocol or runset. Select the check box to require the user to enter a value for the field. Clear the check box if the user can start the protocol or runset without entering a value for the field. *Note:* If an Experiment ID control is not marked as Mandatory, the form's user has the option of leaving it blank. In this case, no experiment data will be generated for a protocol run from the form.
JavaScript variable	The JavaScript variable that will be associated with the control. Type any string that qualifies as valid syntax for a JavaScript variable name. For details on using variables, see "Using simple variables" on page 93.
Always use global context	Specify the context in which to execute the script: • *Global context*. Select the check box to use the global context. For example, if the button is to be pressed while no protocol is running, you must use the global context. **IMPORTANT** Any protocol that uses the global context and contains JavaScript variables of the same name will be affected by changes you make to this variable, even if this form's **Use global context for variables** check box is not selected. • *Context of the running protocol*. Clear the check box to use the protocol's private context.

4 *Optional*. Use the Static text control to add a caption to the Experiment ID control. For details, see "Static text control properties" on page 216.

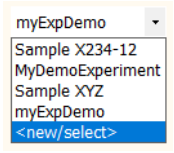
Experiment ID droplist

You can use the Experiment ID droplist on the form instead of the combined Select Experiment ID button and Experiment ID control. If you include the droplist, the form's user can select an ID from the list of previously created experiments or create a new experiment and then select it using this option.

The droplist includes:

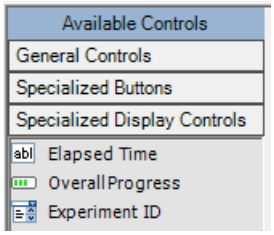
- List of every experiment ID that has an open or not yet used status in the Experiments database.
- A <new/select> option at the bottom of the list that opens the Experiments Editor.

Figure Droplist example



To add the Experiment ID droplist to the form:

1 In the **Form Designer** window or **Editing<*form name*>.VWForm** window, under **Available Controls**, click **Specialized Display Controls**, and then drag the **Experiment ID droplist** to the form canvas area.



Note: To turn off the autogrid, press ALT while dragging the control into position.

2 In the **Form Canvas** area, click the control so that a resizing box appears around the border  .

To resize the control, drag one of the sizing handles.

3 Specify the following properties.

Property	Description
Control	The descriptive name associated with the control. The Control name does not appear on the form. Use a functionally descriptive name so that references to this control are recognizable on audit trail reports. For details, see "About the naming of form controls" on page 184.
Font size	The size of the text that this control will display after the form's operator selects the experiment ID.
Font color	The color of the text that this control will display after the form's operator selects the experiment ID. Click the box to select the color.

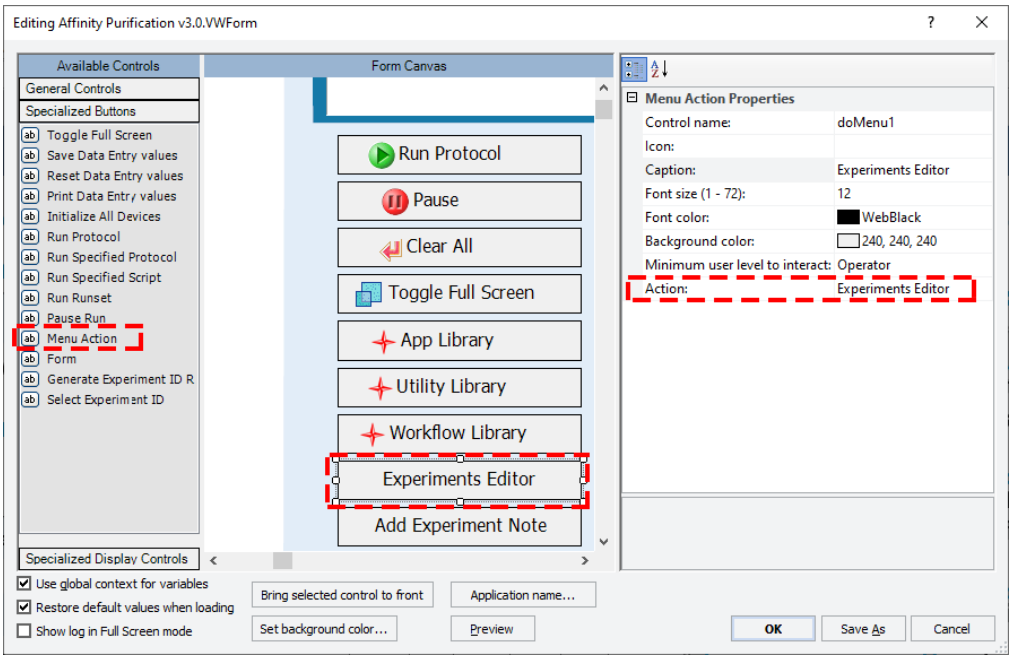
Menu Action buttons for the Experiments Editor

You can include Menu Action buttons with the following actions:

- **Experiments Editor**. Opens the Experiments Editor.
- **Experiments Archive**. Opens the Experiments Archive, which lists all the archived experiment IDs.
- **Add Experiment Note**. Opens the Add Note dialog box.

  Note: Unlike the other Menu Action button options, Add Experiment Note is not available in the Tools menu.

Figure Example of Menu Action button with Action set to Experiments Editor



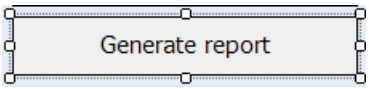
For details on the Menu Action properties, see "Menu Action button properties" on page 196.

Adding Generate Experiment ID Report button

The Generate Experiment ID Report button is a specialized button that creates a report for the experiment ID selected on the form. When a form's user clicks this button, the Save As dialog box opens and allows the user to save the report (.pdf) for the selected experiment at the specified location.

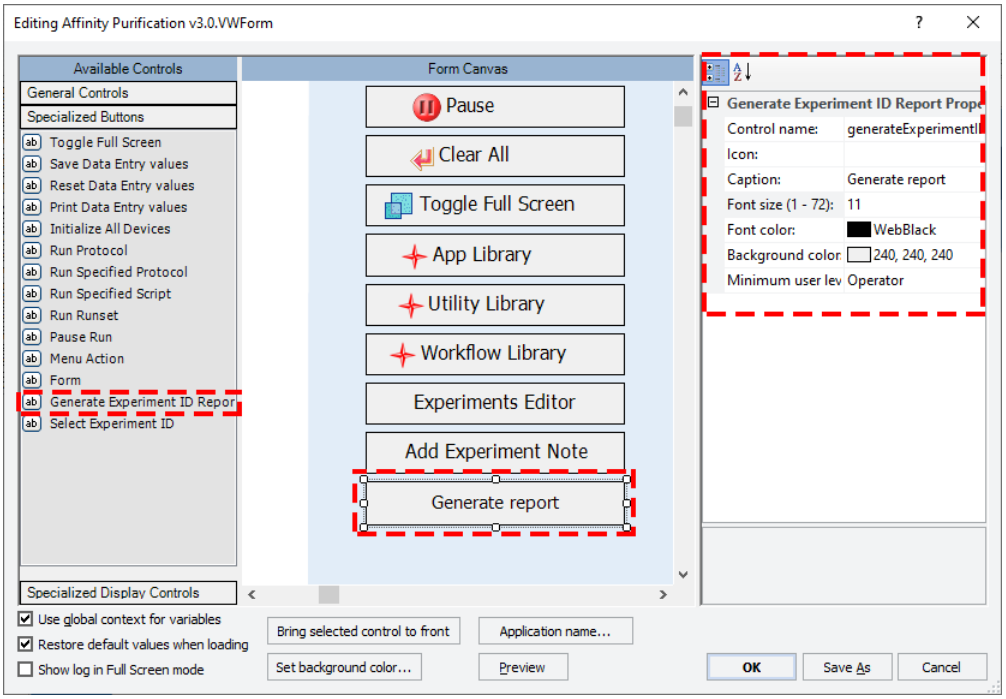
To add the Generate Experiment ID Report button to the form:

1. In the **Form Designer** window or **Editing<*form name*>.VWForm** window, under **Available Controls**, click **Specialized Buttons**, and then drag the **Generate Experiment ID Report** button to the form canvas area.
2. In the **Form Canvas** area, click the control so that a resizing box appears around the border.



To resize the control, drag one of the sizing handles.

Figure Example of Generate Experiment ID Report button



3 Specify the property values. For details, see "Properties specialized buttons have in common" on page 195.

Related information

For information about...	See...
Experiment IDs, Experiments database, and generating reports	*VWorks Automation Control Setup Guide*
Setting up a protocol to track experiment data	"Configuring a protocol to track experiment data" on page 260
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using the experimentTagsDB object in a protocol	"Experiments database objects" on page 111
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Configuring a protocol to track experiment data

This topic describes how to configure a protocol associated with a form that is configured to track experiment data for the Experiments database.

If your protocol is not configured to work with a form, see "Experiments database objects" on page 111 for examples of how you can use JavaScript to configure a protocol to track experiment data.

About the Step Started/Completed task

The Step Started/Completed task (Step Started/Completed) is used to mark a given Step (task sequence) in a protocol as having been started or having completed. Steps are used to track which tags are relevant to a given application run, and which have not had the underlying part of the protocol executed yet.

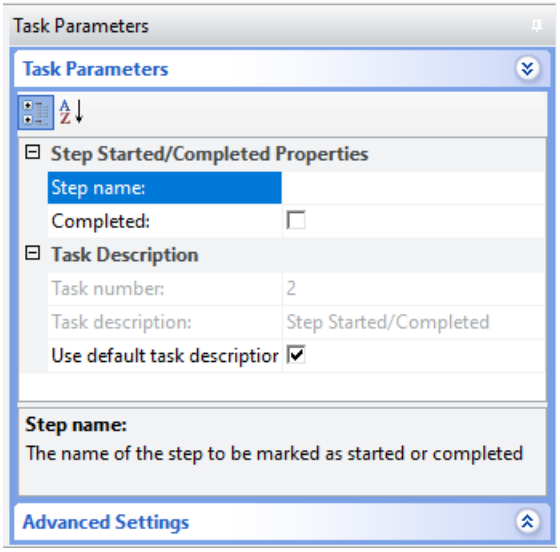
Task is available for...	Task is available in...
Protocols with a connection to the Experiments database	Startup protocol Main Protocol Bravo Subprocess Cleanup protocol

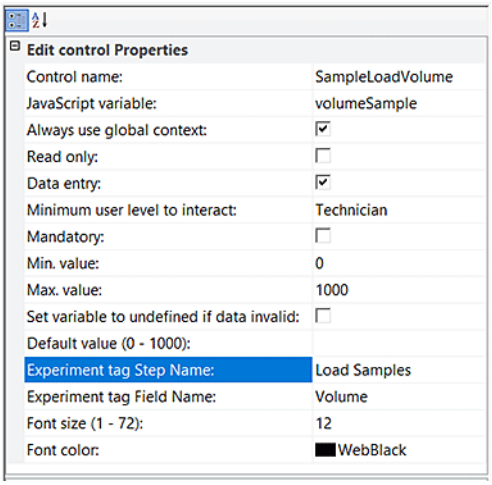
Requirements

The Step Started/Completed task requires a connection to the Experiments database. For instructions, see the *VWorks Automation Control Setup Guide*.

Task parameters

In the protocol, add the Step Started/Complete task at the beginning and end of each step to be tracked in the protocol. After adding the Step Started/Completed task at the desired points in the protocol, set the following parameters in the **Task Parameters** area.



Parameter	Description
Step name	The name of the protocol (application) step for which this data is relevant. Enter the same value that appears in the Experiment tags Step Name property of all controls on the form that are associated with this step. The following figure shows an example of this property for a form control.  **IMPORTANT** If the Step name property is blank in a Step Started/Completed task, an error will occur when you compile the protocol.
Completed	Clear the check box if this instance of the Step Started/Completed task is placed at the beginning of a step to be tracked in the protocol. Select the check box if this instance of the Step Started/Completed task is at the end of a step to be tracked in the protocol.

Example

Goal

Use the Experiments database to track the parameter settings involved in preparing a microplate for downstream analysis.

Implementation

A form is configured to run the protocol for which data is to be recorded in the Experiments database. The form is configured to include:

1. Experiments ID control
2. Application name

   In this example, Affinity Purification is the name of the application.
3. Each form control to be tracked has the following properties defined:
   - **Experiment tag Step Name**. The following figure shows an example for the Load Samples step.

– **Experiment tag Field Name**. The following figure shows an example for the Volume control (Edit control).

In the protocol, the steps to be tracked are bracketed by:

- **Step Started/Completed task at beginning of step**. This instance has a cleared Completed check box.
- **Step Started/Completed task at the end of the step**. The Completed check box is selected for this instance of the task.

Note: A given Step can be Started only once and Completed only once, and it cannot be Completed before it is Started. The software will not execute a Step Started/Completed task that breaks this rule.

Figure Example of Volume control properties for a Load Samples step on a form

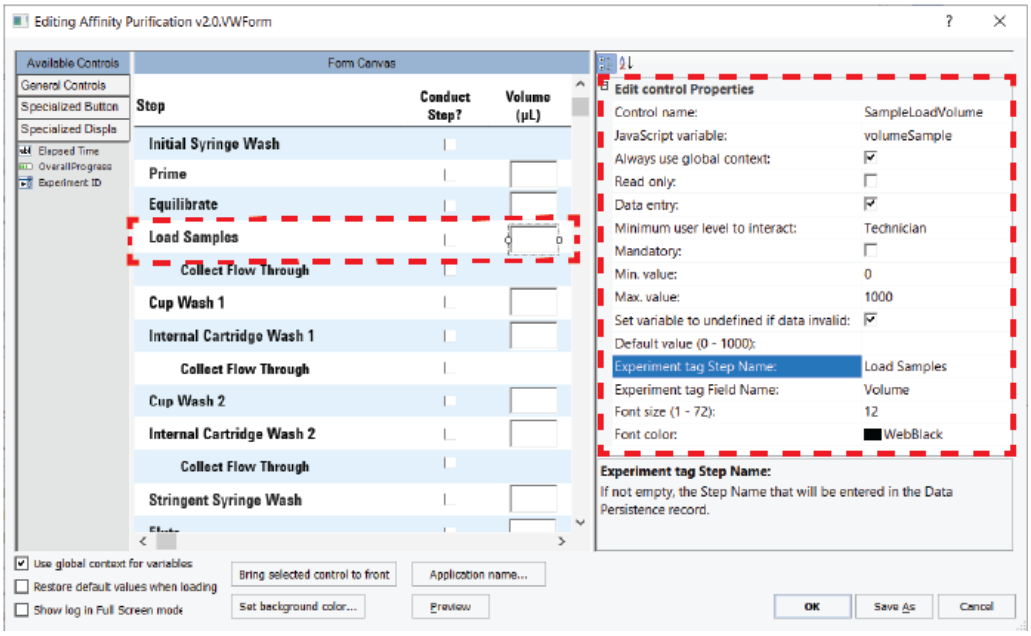
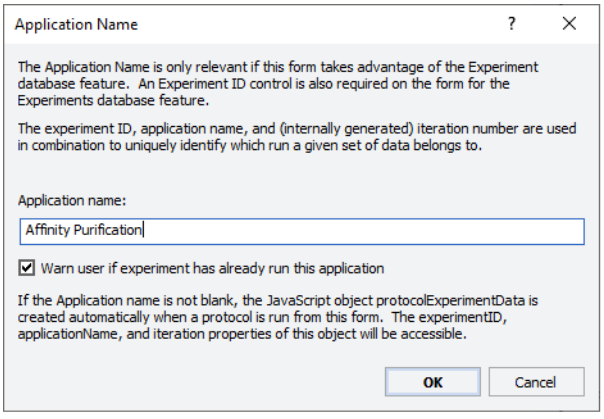


Figure Example of the Application Name dialog box



The following figures show examples of the Start/Stop Completed tasks at the beginning and end of a Load Sample step.

Figure Example of Step Started/Completed task marking the beginning of a protocol step

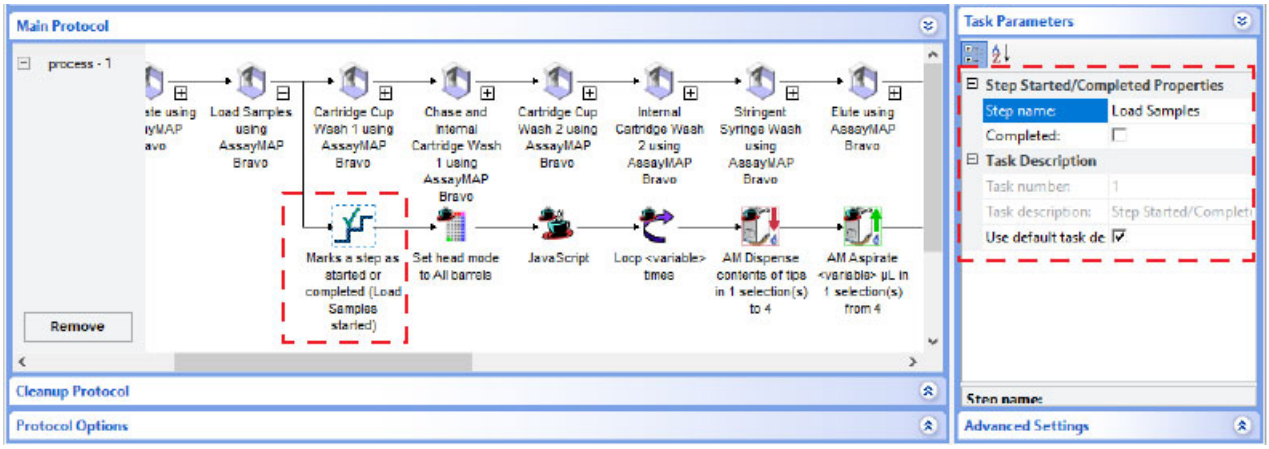
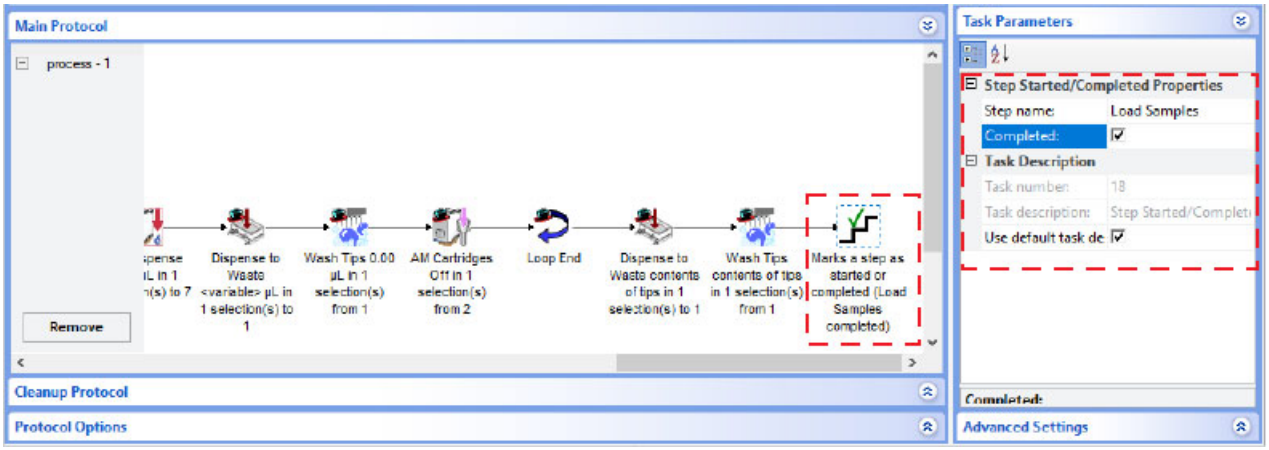


Figure Example of Step Started/Completed task marking the end of a protocol step



To create a report (.pdf file) for the experiment ID that includes this protocol after the protocol is run, do one of the following:

- In the form, click the **Generate Experiment ID Report** button, if included.
- In the **VWorks** window, click **Tools > Experiments Editor**, select the experiment ID, and then click **Report for selected.**

Related information

For information about...	See...
Experiment IDs, Experiments database, and generating reports	*VWorks Automation Control Setup Guide*
Setting up a form to track experiment data	"Configuring a form for experiment IDs and Experiments database" on page 249
Workflow for creating or editing a form	"Workflow for creating or editing a form" on page 181

For information about...	See...
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using the experimentTagsDB object in a protocol	"Experiments database objects" on page 111
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Using macros to create protocols

This chapter describes the VWorks macros that you can use to help simplify and expedite the protocol writing process. You must be an administrator or technician access to create macros, and you should have an understanding of how to create VWorks protocols.

This chapter contains the following topics:

- "About protocol macros and the macro library" on page 266
- "Adding macros to the macro library" on page 269
- "Inserting macros in protocols" on page 272
- "Editing a macro" on page 276
- "Adding parameter variables and scripting to a macro" on page 282
- "Managing macro libraries" on page 285

About protocol macros and the macro library

About this topic

This topic describes the VWorks macros and the macro library that you can create to help simplify and expedite the protocol writing process. You must be an administrator or technician to create macros and protocols.

Macros defined

A VWorks macro is a collection of protocol tasks and associated task parameters grouped together in an abbreviated form. You can insert the macro easily at other places within the same protocol or throughout other protocols where the same task sequence is required.

A macro eliminates the need to recreate a repeated task sequence within the same protocols or subsequent protocols that you write. Therefore, protocol writing can be easier, faster, and less error-prone.

Suppose you want to create a protocol that has multiple wash routines, where each wash routine consists of a sequence of Tips On, Aspirate, Dispense, and Tips Off tasks. You could do the following using a macro:

1. Create the wash routine task sequence.
2. Highlight the task sequence and add it as a macro to the macro library.
3. Set custom variables at the macro level, such as the volume or number of times to wash, so that the parameters could be scripted before running the protocol.

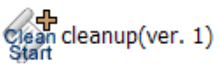
   For instance, the variables could be exposed on a protocol form that the operator is permitted to change at run time.
4. Insert the macro at the other points in the protocol where the wash routine is required.

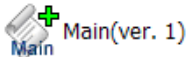
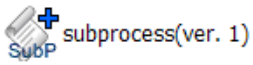
For the subsequent protocols that you create, which require the wash routine, you can insert the macro from the macro library. For each instance of the macro, you should verify that the parameter values are set correctly, such as Location, plate and Location, location.

Later, if you want to modify the macro, the change can be replicated to all the other instances of the macro in an open protocol.

Types of macros

You can create the following types of macros in the VWorks software:

Macro	Description
Startup or cleanup macro 	Available for use in the Startup Protocol area and the Cleanup Protocol area.

Macro	Description
Main process macro 	Available for use in the Main Protocol area only within the main process.
Subprocess macro 	Available for use in the Main Protocol area only within a subprocess.

Macros have the following requirements:

- All tasks in the macro must be contiguous in the protocol.
- A macro can contain subprocess or process tasks, but not both.
- A macro cannot contain another macro.

Macro library

The macro definitions are stored in a macro library (.mlb file). Although different versions of a given macro may be in use in the protocols, the software uses only one macro library for a given VWorks session.

- To specify the macro library file path, see "Viewing and setting file directory paths" on page 299.
- To save a copy of a macro library, see "Managing macro libraries" on page 285.

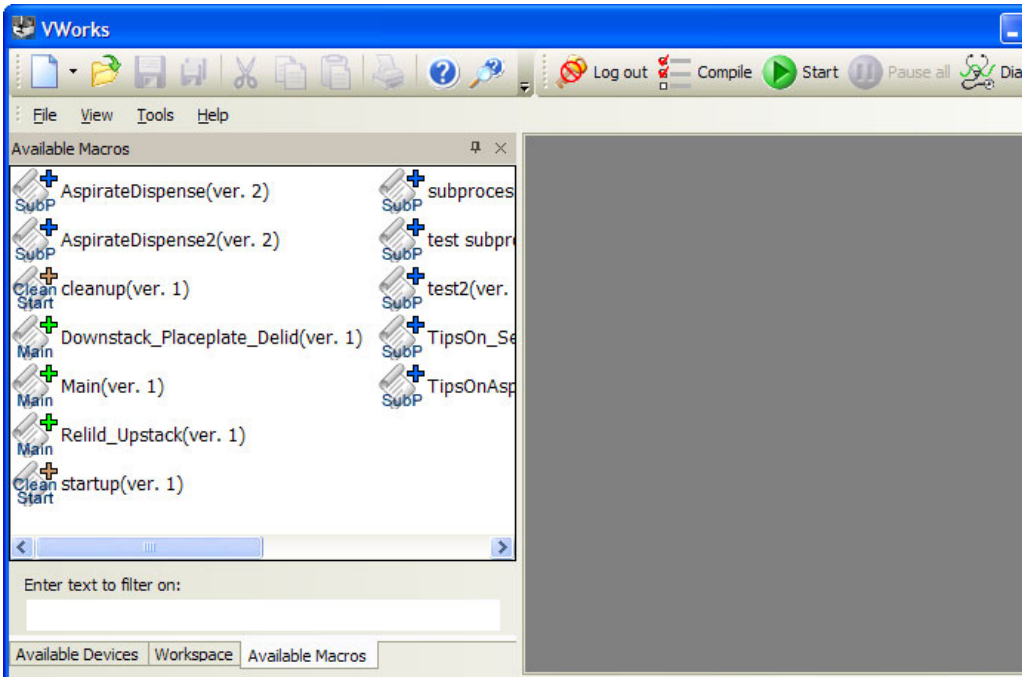
The software stores the macro libraries in the VWorks Project folder, for example,

/VWorks Projects/VWorks/Miscellaneous/Macro Library/MacroLibrary.mlb

About available macros

The Available Macros area in the VWorks window displays the contents of the specified macro library.

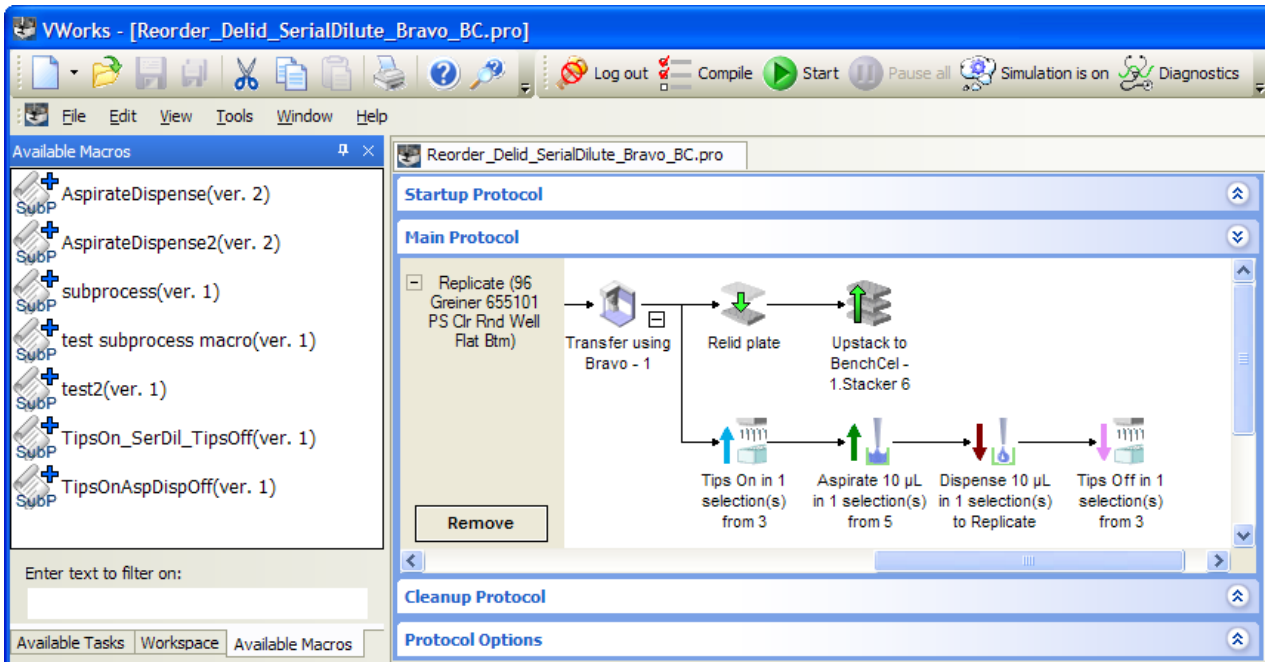
Figure Available Macros area displaying the contents of the macro library



The Available Macros area lists the macros alphabetically.

- If no protocol is open, the macro list includes all types of macros.
- If a protocol is open, a subset of the macro library appears in the list depending on which area of the protocol tab is active and which devices are used in the protocol. For example, in the following figure only the subprocess macros for the Bravo Platform appear.

Figure Available Macros area displaying only the subprocess macros



Related information

For information about...	See...
Creating macros	"Adding macros to the macro library" on page 269
Editing macros and macro version control	"Editing a macro" on page 276
Using a macro in a protocol	"Inserting macros in protocols" on page 272
Exporting and importing the macro library	"Managing macro libraries" on page 285
Exporting and importing protocols	"Exporting and importing protocols and associated records" on page 703
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Working with JavaScript	"Creating a protocol: advanced topics" on page 89

Adding macros to the macro library

About this topic

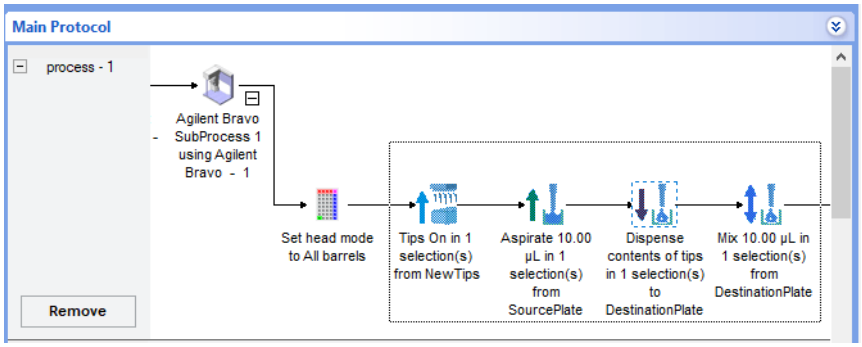
This topic describes how to create macros and add them to the macro library and how to delete macros.

The procedures in this topic assume that a protocol task sequence that you want to replicate as a macro is already created. For details on how to set up tasks and create a protocol, see "Workflow for creating a basic protocol" on page 22.

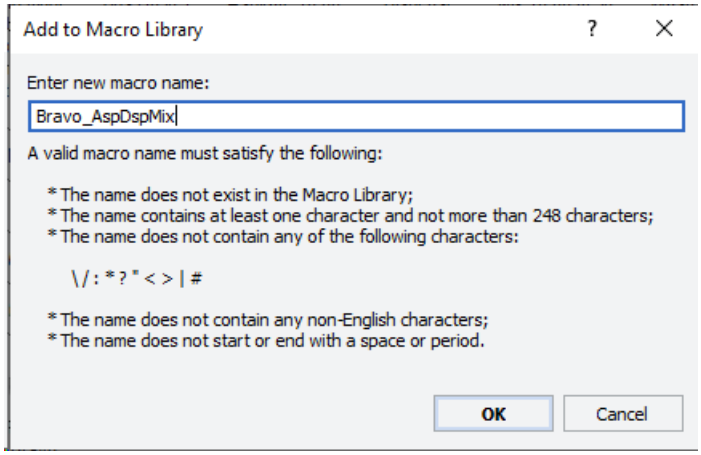
Creating a macro and adding it to the macro library

To create a macro and add it to the macro library:

1 In the protocol area, drag the pointer around the task or series of tasks that you want to include in the macro. Make sure that only the desired tasks are highlighted.



2 Right-click the selection, and then choose **Add to macro library** from the shortcut menu. The Add to Macro Library dialog box opens.



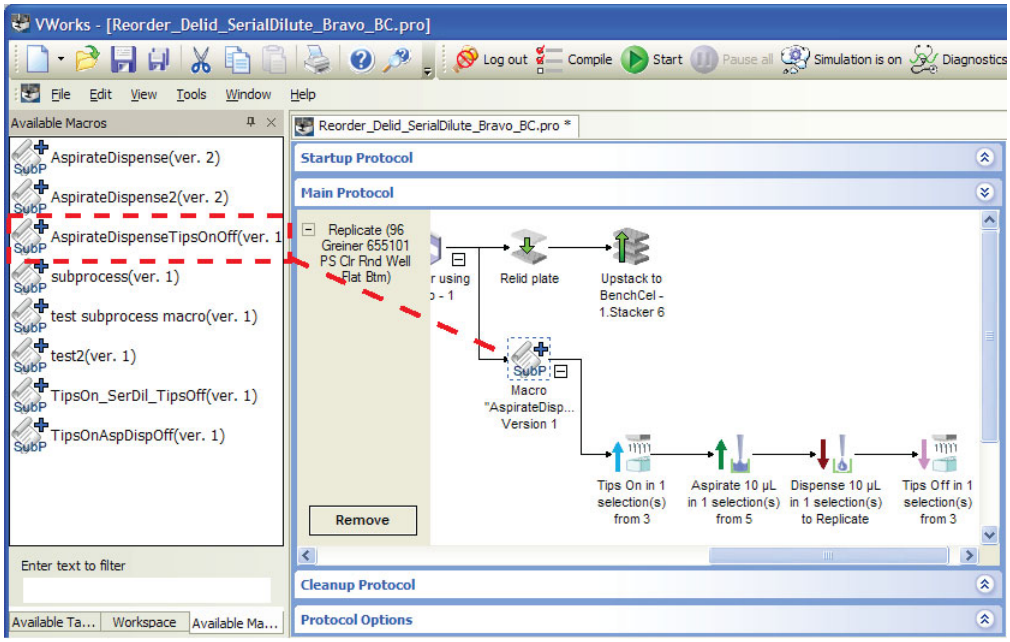
3 Type a name for the macro, and click **OK**. A descriptive name can be more helpful for other macro users than a generic name. A version number is appended to the macro name, starting with version 1.

4 *VWorks Plus only.* If an audit trail is being logged for the macro library, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

IMPORTANT Although you can rename a macro in the macro library, doing so removes any association to instances of the macro with the former name that are already in use in a protocol. Avoid changing the macro name after the macro has been used in a protocol.

5 In the protocol area, expand the macro icon to view and to verify the macro contents.

The corresponding macro icon appears in the Available Macros area and in the protocol. Any future edits of the macro will increment the version number by 1.



Note: To associate a custom icon or custom parameters with the macro, see "Editing a macro" on page 276.

Adding a copy of a macro or saving the macro by a different name

You can create a copy of a macro to add to the macro library. For example, you might want to copy a macro and use the copy as the starting point for creating a similar but slightly different macro.

To create a copy of a macro from the Available Macros tab:

1. In the **Available Macros** tab, right-click the macro icon, and then choose **Make a copy** from the shortcut menu.
2. In the **Create Macro Copy** dialog box, type a new name for the copy, and click **OK**. The new macro icon appears in the Available Macros area.
3. *VWorks Plus edition only.* If an audit trail is being logged for the macro library, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

To save a copy of a macro from a protocol:

1. In the protocol, right-click the macro icon, and then choose **Save macro as** from the shortcut menu.
2. In the **Add to Macro Library** dialog box, type a new name for the copy, and click **OK**. The new macro icon appears in the Available Macros area.
3. *VWorks Plus edition only.* If an audit trail is being logged for the macro library, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

Deleting a macro from the macro library or a protocol

You can delete a macro from the macro library or a macro instance from a protocol:

- *Deleting a macro from the macro library*. In this case, instances of the deleted macro can still remain in your protocols. However, the instances of the deleted macro no longer have version control or any association with other instances.
- *Deleting an instance of a macro from a protocol*. In this case, only the selected instance is deleted from the protocol. The macro remains in the macro library and in other protocols where it is used.

  Note: In a protocol where you no longer want a macro, yet you still want to retain a macro's contents (task sequence), you can revert the macro. See "Reverting an instance of a macro to an ungrouped task sequence" on page 274.

To delete a macro from the macro library:

1. In the **Available Macros** area, right-click the macro icon, and then choose **Delete** from the shortcut menu. The macro icon disappears from the Available Macros list.
2. *VWorks Plus only.* If an audit trail is being logged for the macro library, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

To delete an instance of a macro from a protocol:

In the protocol, right-click the macro icon, and then choose **Delete** from the shortcut menu. The macro icon and macro task contents disappear from the protocol.

Related information

For information about...	See...
Editing macros and understanding macro version control	"Editing a macro" on page 276
Adding a custom icon to the macro	"Associating a custom icon with the macro" on page 278
How to expose task parameter variables at the macro level	"Adding parameter variables and scripting to a macro" on page 282
Using a macro in a protocol	"Inserting macros in protocols" on page 272
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22

Inserting macros in protocols

About this topic

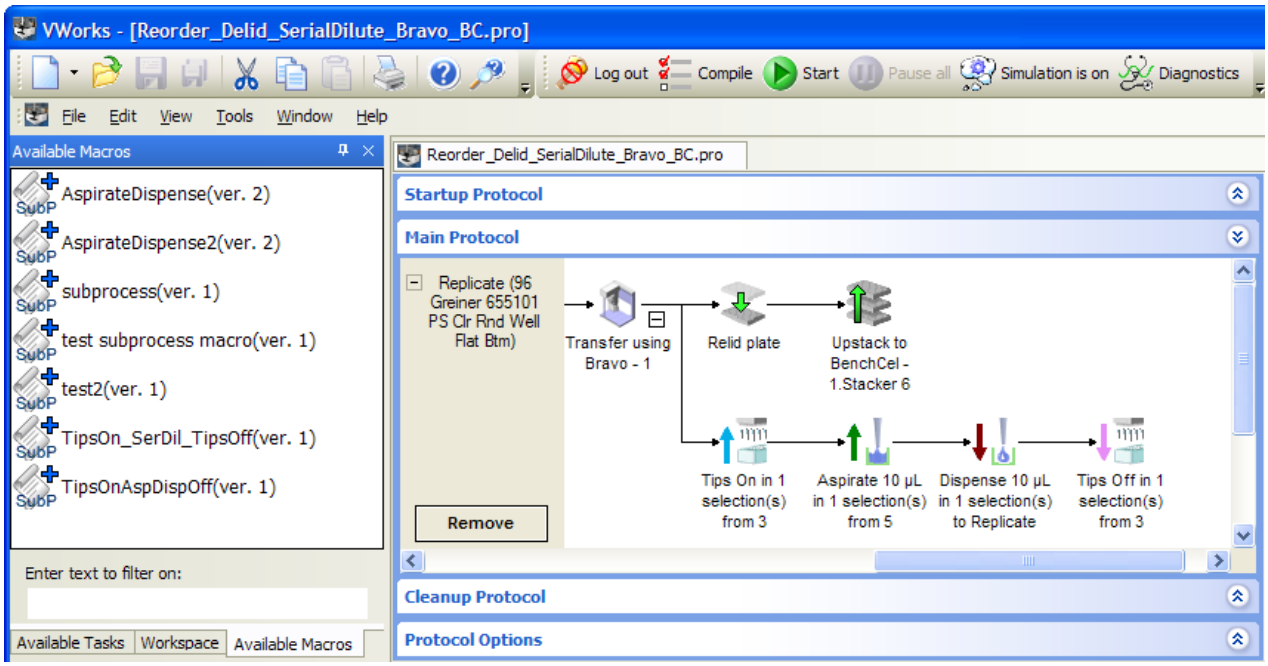
This topic describes how to insert macros into the protocols that you are creating. You must be an administrator or technician to create protocols and work with macros.

Viewing and filtering the list of macros

In the VWorks window, the Available Macros area lists only the macros from the macro library that are relevant for the protocol type (startup, cleanup, main process, or subprocess) and associated devices selected in the protocol tab.

In the following figure, a Bravo subprocess appears in the Main Protocol area and the Available Macros area lists only subprocess macros for the Bravo Platform.

Figure Available Macros area displaying subprocess macros for the Bravo Platform



You can filter the Available Macros list even further.

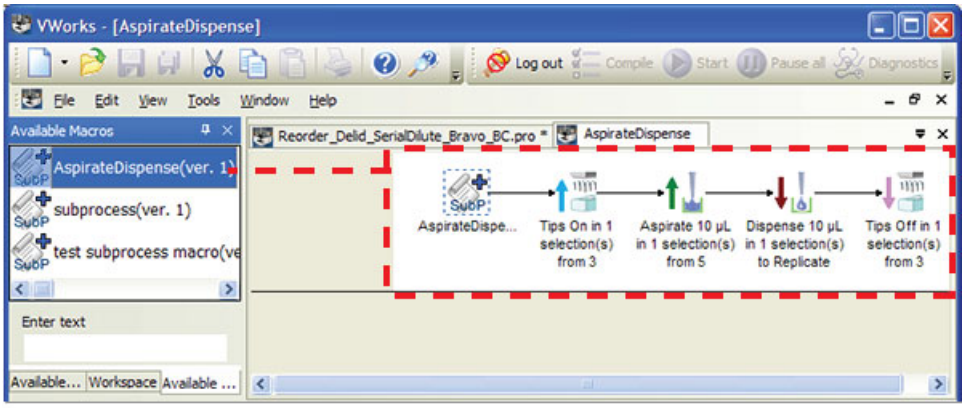
To view and filter the list of macros in the macro library:

1 In the VWorks window, choose **View > Available Macros** to display the Available Macros area if not already displayed.

2 In the **Available Macros** area, type the text in the **Enter text to filter on** box that you want to use to filter the list of macro names. As you type each character, the Available Macros list changes to meet the filter requirement.

For example, if you type *wash*, only the macros that include the name *wash* remain in the Available Macros list.

Note: To preview a macro's contents, right-click the macro icon in the **Available Macros** tab, and then choose **Edit** from the shortcut menu. A macro editor tab appears in protocol area, as the following figure shows. To close the macro editor tab, click the corresponding **x** button.

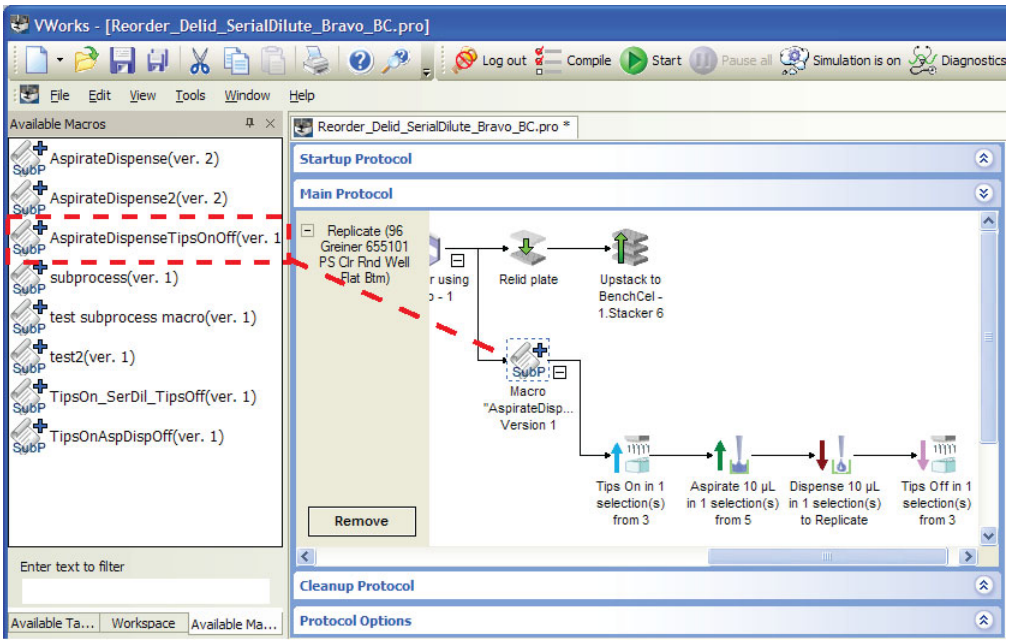


Inserting a macro in a protocol

To insert a macro in a protocol:

1. Click the position in the protocol where you want to insert the macro.
2. In the **Available Macros** area, double-click the macro icon. The selected icon appears in the protocol at the location you selected.

   Alternatively, you can drag the macro from the **Available Macros** area into position in the protocol.
3. To view the contents of the macro within the protocol, expand the macro icon.



4. Verify that the task parameter settings are correct for the new instance of the macro, including Location, plate and Location, location.

Reverting an instance of a macro to an ungrouped task sequence

If you have an instance in a protocol where you no longer want a macro, yet you still want to retain a macro's contents (task sequence), you can revert the macro.

To revert an instance of a macro:

In the protocol area, right-click the macro icon, and then choose **Ungroup macro**.

The macro icon disappears from the protocol, and the tasks from the macro remain inserted in the protocol position where the macro formerly resided.

Related information

For information about...	See...
Adding macros and removing macros from the macro library	"Adding macros to the macro library" on page 269
How to edit macros and macro version control	"Editing a macro" on page 276
Exporting the macro library	"Managing macro libraries" on page 285
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Editing a macro

About this topic

You must be an administrator or technician to edit a macro.

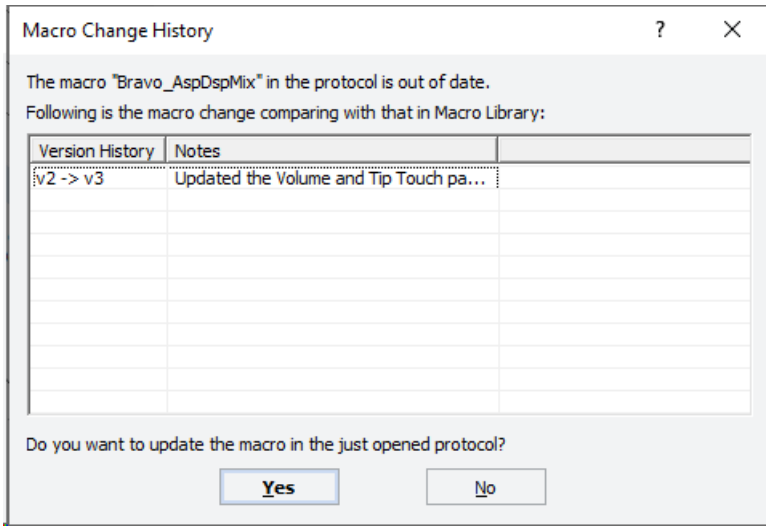
Editing options and macro version control

The macro library stores the latest version of a macro that has been edited in the macro editor tab. You can use different versions of a given macro in your protocols, but the different versions reside only in the protocols.

You can edit the VWorks macros in the following ways:

- *Editing a macro from the macro library in the macro editor tab*. Replaces the version of the same name in the macro library, incrementing the version number by 1.

  Later, if you open a protocol that has an earlier macro version than the macro in the macro library, the following Macro Change History dialog box opens. You can choose whether to update the macros in the protocol to match the definition in the macro library.



- *Editing a macro directly in the protocol tab*. Updates the macro in the protocol only. Macros in the macro library remain unchanged. Instead, you have the following options:
  - Keep a derivative of the definition from the macro library in the selected protocol.
  - Replace the modified instances in the protocol with the definition from the macro library.
  - Add the modified definition under a new name in the macro library.

  The software does not maintain a change history of the macros that are edited in the protocol tab. However, you can easily update all instances of a macro in a protocol to match the version of the same name in the macro library. See "Updating instances of a macro to match the version in the macro library" on page 280.

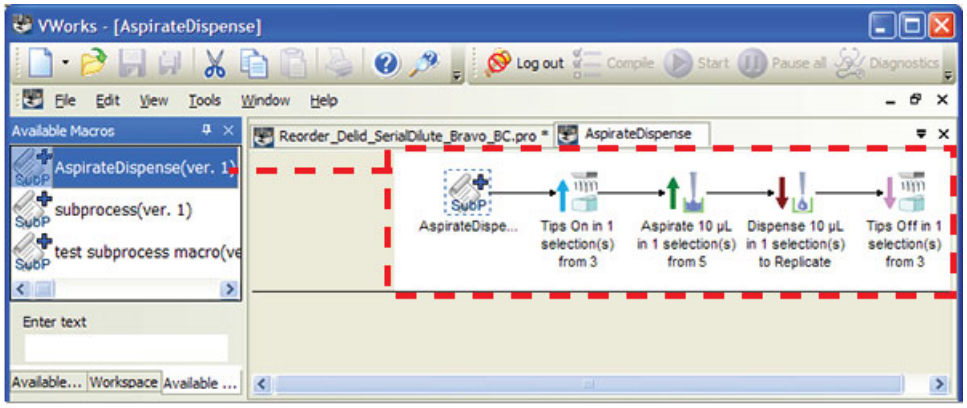
Editing a macro from the library in the macro editor tab

To edit a macro from the library in the macro editor tab:

1 In the **Available Macros** tab, right-click the macro icon, and then choose **Edit** from the shortcut menu.

A macro editor tab opens in protocol area and displays the contents of the macro as the following figure shows.

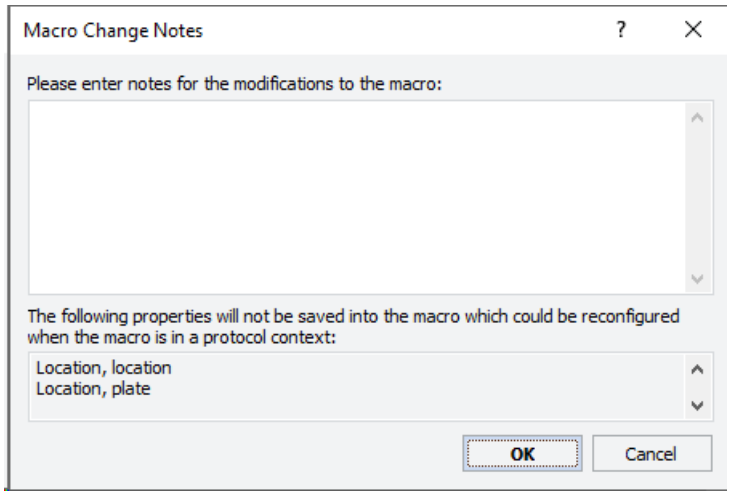
Figure Macro editor tab



2 To add or delete a task:

- *Add a task*. Either drag the task icon from the **Available Tasks** area to the macro editor tab, or double-click the task icon in the **Available Tasks** area to add the task to the macro editor tab.
- *Delete a task*. In the macro editor tab, right-click the task icon, and then choose **Delete** from the shortcut menu.
- *Change a task's parameter values*. Click the task icon in the macro editor tab. In the **Task Parameters** area, edit the task parameters as you typically would when creating a protocol.
- Associate a custom icon with the macro. See "Associating a custom icon with the macro" on page 278.
- Associate custom parameters with the macro. See "Adding parameter variables and scripting to a macro" on page 282.

3 Choose **File > Save**. The Macro Change Notes dialog box opens:



4 Type a description of your edits, and then click **OK**.

Later, if you open a protocol containing a different version of this macro, the software displays this change history for reference.

5 *VWorks Plus only.* If an audit trail is being logged for the macro library, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

6 When the message appears, asking if you want to update the macro in the protocol:

- Click **Yes** to update all instances of the macro with the same name in the specified protocol, regardless of version number. The software increments the macro version number by 1 in the macro library.
- Click **No** to increment the macro version number by 1 in the macro library only. Any previous versions of the macro in the protocol with the same macro name remain in the protocol intact.

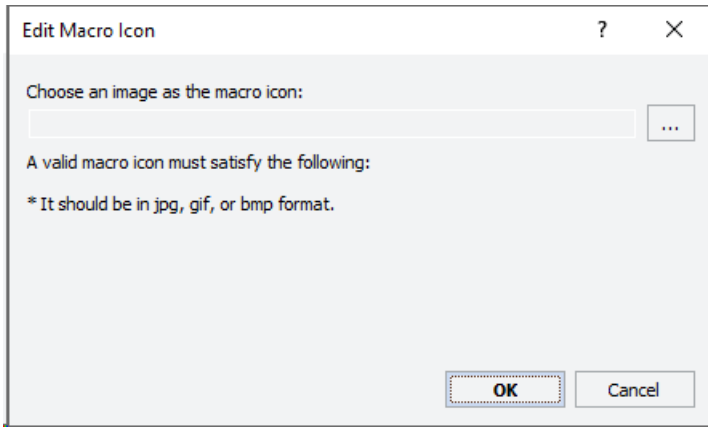
If any unopened protocols also contain the previous version of the macro, you have the option to update the macros when you open each protocol.

Associating a custom icon with the macro

To associate a custom icon with the macro:

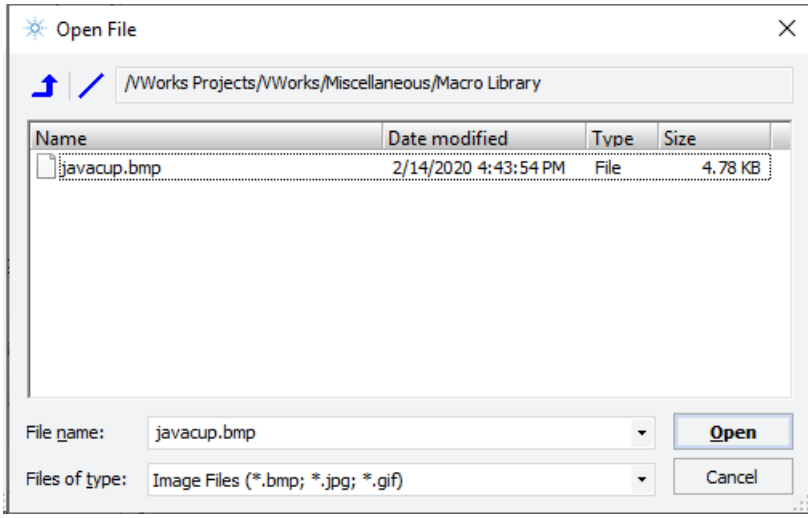
1 Ensure that the image file (.bmp, .jpg, or .gif) has been imported into VWorks file storage. For details, see "Importing and exporting miscellaneous files" on page 49.

2 In the macro editor tab or protocol area, right-click the macro icon, and then choose **Add Custom Icon** from the shortcut menu. The Edit Macro Icon dialog box opens.



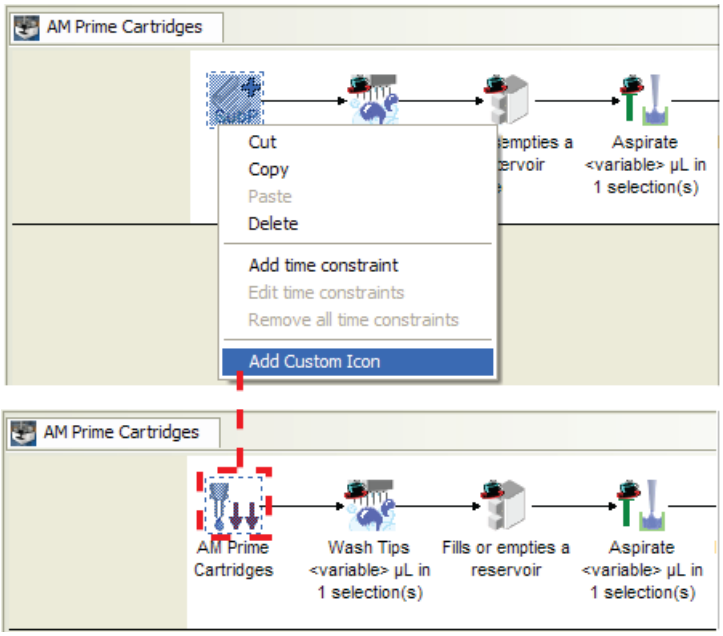
3 In the **Edit Macro Icon** dialog box, click the ... button, the Open File dialog box opens.

4 In the **Open File** dialog box, select the image file, and then click **Open**.



5 In the **Edit Macro Icon** dialog box, click **OK**. The selected image replaces the macro icon in the macro editor tab.

Figure Macro before (top) and after (bottom) adding custom icon in macro editor tab



Editing a macro directly in the protocol tab

Macros in the macro library remain unchanged when you edit an instance of a macro in the protocol.

To edit a macro in the protocol tab:

1 In the protocol, expand the macro icon so that you can view the macro task contents.

2 Edit the tasks as desired:

- *Add a task*. Drag the task icon from the **Available Tasks** area to the macro in the protocol, or double-click the task icon in the **Available Tasks** area.

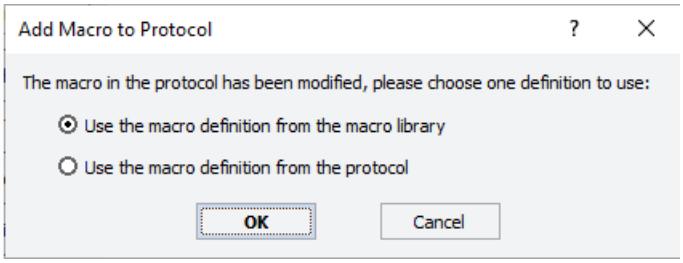
Alternatively, you can copy a task icon from elsewhere in the protocol. To do this, right-click the task icon, and then choose **Copy**. Click the target location in the macro, right-click, and then choose **Paste**.

- *Change task parameter values*. Click the task icon. Edit the values in the **Task Parameters** area as you typically would when creating a protocol.
- *Delete a task*. Right-click the task icon, and then choose **Delete** from the shortcut menu.
- Associate a custom icon with the macro. See "Associating a custom icon with the macro" on page 278.
- Add custom parameters and scripting to the macro. See "Adding parameter variables and scripting to a macro" on page 282.

3 Do one of the following:

- *To rename the edited macro and add it to the macro library*. In the protocol, right-click the icon of the edited macro, and then choose **Save macro as**. The Add to Macro Library dialog box opens.
- *To keep the edited macro definition in the protocol only.* Choose **File > Save**. The changes are applied to all instances in the protocol that have the same name and version number. The name of the edited instance changes to "*macro_name* Derived from Version n".

  Note: If you subsequently add an additional macro of the same name from the macro library to this protocol, the Add macro to protocol dialog box opens. You can choose which definition to use for all the macros in the protocol that have the same name.



Updating instances of a macro to match the version in the macro library

If a protocol contains a derivative version of a macro or a different version than one of the same name in the macro library, you can choose to update all instances in the open protocol to the definition in the macro library or to keep the definition from the protocol.

To update a macro in the protocol tab to the current version in the macro library:

In the protocol tab, right-click the macro icon, and then choose one of the following:

- **Update this macro.** Replaces the selected macro instance in the open protocol with the definition from the macro library.
- **Update all revisions of this macro**. Replaces all macro instances of the same name with the definition from the macro library, regardless of the version number of the other macro instances in the protocol.

Renaming a macro in the macro library

CAUTION **Renaming a macro in the macro library removes any association to the macro instances of the former name that are in the protocols.**

To rename a macro in the macro library:

1. In the **Available Macros** tab, right-click the macro icon, and then choose **Rename** from the shortcut menu.
2. In the **Rename Macro** dialog box, type the new name and click **OK**.
3. *VWorks Plus only.* If an audit trail is being logged for the macro library, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

Related information

For information about...	See...
Adding macros and removing macros from the macro library	"Adding macros to the macro library" on page 269
Using macros in protocols	"Inserting macros in protocols" on page 272
Exporting the macro library	"Managing macro libraries" on page 285
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22
Using JavaScript	"Creating a protocol: advanced topics" on page 89

Adding parameter variables and scripting to a macro

About this topic

This topic describes the following:

- "Adding macro parameter variables" on page 282
- "Filtering the macro parameters" on page 283
- "Adding JavaScript to a macro" on page 283

This topic assumes that you have an understanding of how to create VWorks protocols, including how to do the following:

JavaScript tasks	For details, see...
Define JavaScript variables in the VWorks software	"Using simple variables" on page 93
Write programs in JavaScript in the VWorks software	"Using JavaScript" on page 100

Adding macro parameter variables

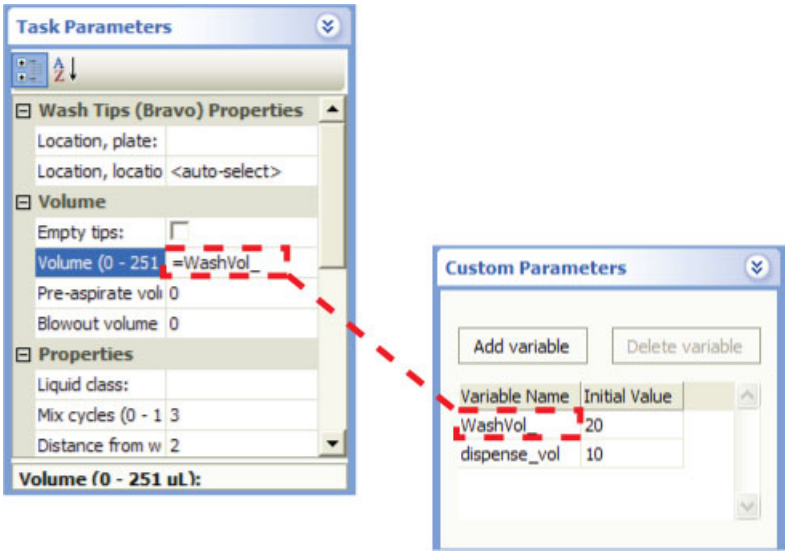
You can expose parameters for individual tasks at the task macro level. For example, you might want to edit the parameter values of tasks contained in the macro without having to open the macro and then open each task to access the task parameters.

To add task parameter variables to the macro:

1. In the **Available Macros** tab, right-click the macro icon, and then choose **Edit** from the shortcut menu.

   A macro editor tab opens in protocol area and displays the contents of the macro.
2. In the macro editor tab, click the macro icon. The Custom Parameters area replaces the Task Parameters area next to the protocol area.
3. Click **Add Variable**. A row appears in the Variable Name table.
4. In the new row, click the **Variable Name** field, and then type the exact same JavaScript variable that appears in the Task Parameters area for the given task parameter.

   For example, a macro contains multiple Wash Tips tasks for which you are using the same volume. In the corresponding Task Parameters for each task, you define the Volume parameter as a variable, such as `=WashVol_`. You use the same variable, `WashVol_` as the variable name in the macro's custom parameters. You can change the volume specified for both tasks by editing the variable value in the macro's Custom Parameters area.



5 In the **Initial Value** field of the **Custom Parameters** area, type the value for variable.

For more details on using JavaScript variables for task parameters, see "Using simple variables" on page 93.

6 *VWorks Plus only.* If you are editing a macro in the macro library and an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

Filtering the macro parameters

To show all macro parameters:

In the **Custom Parameters** area, right-click the variables table, and select **Show all**.

To filter the macro parameter list:

1 In the **Custom Parameters** area, right-click the variables table, and select **Show all**.

2 Right-click a cell in the table, and select from the available filtering options:

View	Description
Last filter	Returns the display to when the last filter was applied.
Filter by row	Displays the parameters that have the same value as the parameter in the selected row. For example, if you choose to filter the list based on the initial value of a given parameter. Only the parameters that have the same value as the parameter in the selected row are displayed.
Filter by column	Displays the parameter names that are an exact match of the selected column.

Adding JavaScript to a macro

You can write JavaScript for a macro in a similar way that you can for a task. For example, you could enable a macro for one set of conditions and disable the macro for all other conditions using the `task.skip()` method.

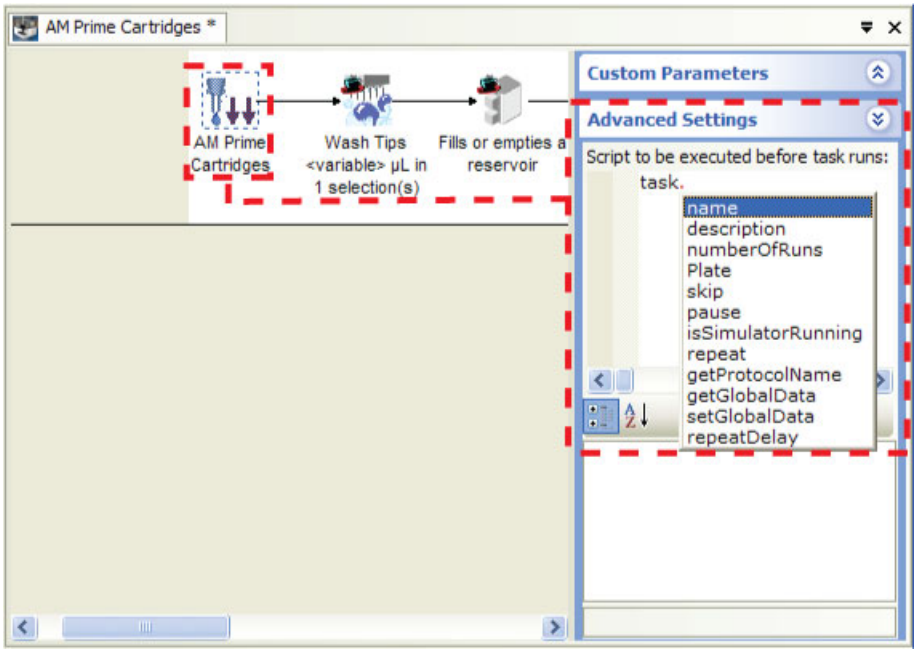
At the macro level, you can use only the shared methods of the task object. For a description of the task object's shared methods, see "task object" on page 126.

To add JavaScript to a macro:

1 In the **Available Macros** tab, right-click the macro icon, and then choose **Edit** from the shortcut menu.

A macro editor tab opens in protocol area and displays the contents of the macro.

Figure Macro editor tab and the Advanced Settings area



2 In the macro editor tab, click the macro icon. The Custom Parameters area replaces the Task Parameters area next to the protocol area.

3 Click **Advanced Settings**.

4 In the **Script to be executed before task runs** box, type your script.

Note: To view the list of available properties for the task object, type `task`. You can select a property from the list that appears.

Related information

For information about...	See...
Adding macros to the macro library	"Adding macros to the macro library" on page 269
Macro version control	"Editing a macro" on page 276
VWorks JavaScript variables	"Using simple variables" on page 93
Advanced Parameters area	"task object" on page 126

Managing macro libraries

To work with macros, you must be an administrator or technician. You can save copies of macros in the following ways:

- *Copy the entire macro library*. Use the following procedure to save a copy of the macro library.
- *Export and import the macros that are part of a protocol*. If you export a protocol (.vzp file), the macros in the protocol are included in the exported .vzp file.

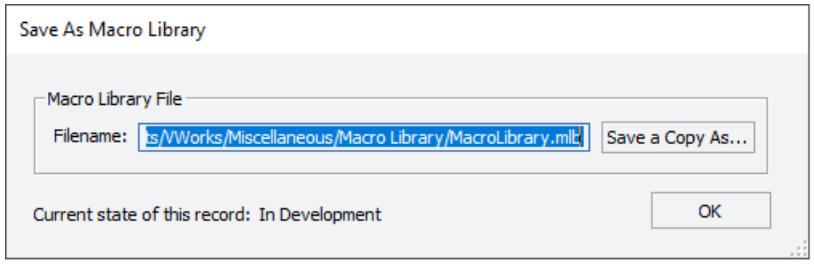
  The Export Wizard also provides the option to export the macro library file.

  For details, see "Exporting and importing protocols and associated records" on page 703.

Saving a copy of a macro library

To save a copy of the macro library:

1 In the VWorks window, click **Tools > Macro Library**. The Save As Macro Library dialog box opens.



2 Under **Macro Library File**, click **Save a Copy As**. The Save File As dialog box opens, which allows you to save the file to OLSS storage.

3 Type a file name, and then click **Save**.

To specify the macro library file path, see "Viewing and setting file directory paths" on page 299.

Related information

For information about...	See...
Exporting and importing macros	"Exporting and importing protocols and associated records" on page 703
Macros and the macro library	"About protocol macros and the macro library" on page 266
Creating macros	"Adding macros to the macro library" on page 269
Using macros to help create protocols	"Inserting macros in protocols" on page 272
Creating a basic protocol	"Workflow for creating a basic protocol" on page 22

For information about...	See...
Working with JavaScript	"Creating a protocol: advanced topics" on page 89

6 Running a protocol

This chapter contains the following topics:

- "Workflow for running a protocol" on page 288
- "Opening a form" on page 289
- "Opening a protocol" on page 291
- "Disabling and enabling a device in the device file" on page 293
- "Viewing and setting file directory paths" on page 299
- "Setting general and view options" on page 301
- "Setting error-handling options" on page 307
- "Setting up email notification" on page 309
- "Starting the protocol run" on page 312
- "Managing runsets" on page 317
- "Monitoring the overall run progress" on page 329
- "Tracking the run progress of instances or devices" on page 330
- "Pausing the run" on page 337
- "Stopping the run" on page 341

Workflow for running a protocol

Step	For this task...	See...
1	Turn on the devices that will be used in the protocol and prepare them for operation.	Device user guide
2	Log in to the VWorks software.	*VWorks Automation Control Setup Guide*
3	Open a form or a protocol.	One of the following: • "Opening a form" on page 289 • "Opening a protocol" on page 291
4	*Optional*. Temporarily disable a device.	"Disabling and enabling a device in the device file" on page 293
5	*Optional*. Set general and view options.	"Setting general and view options" on page 301
6	*Optional*. Set error-handling options.	"Setting error-handling options" on page 307
7	*Optional*. Set up email notification.	"Setting up email notification" on page 309
8	Start the protocol run.	"Starting the protocol run" on page 312
9	*Optional*. Create and manage runsets.	"Managing runsets" on page 317
10	Monitor the run.	"Monitoring the overall run progress" on page 329
11	*Optional*. Pause the run.	"Pausing the run" on page 337
12	*Optional*. Stop the run.	"Stopping the run" on page 341

Related information

For information about...	See...
Preparing devices for a run	Device user guide
Creating protocols	"Workflow for creating a basic protocol" on page 22

Opening a form

About this topic

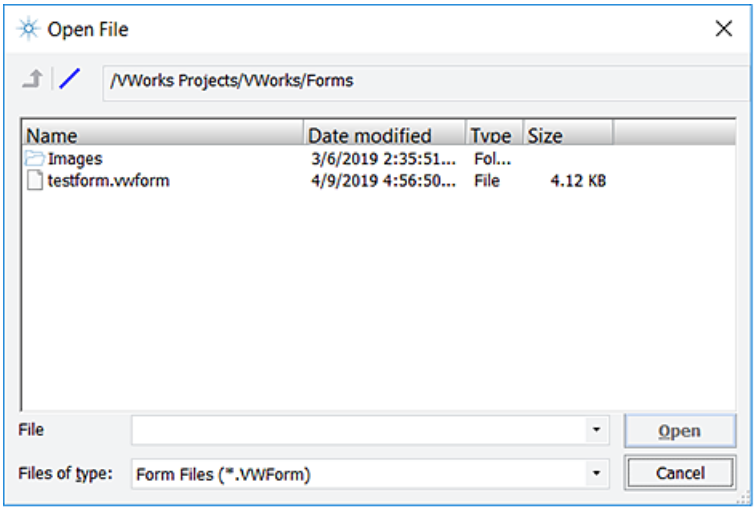
The instructions in this topic assume that you have designed the form and you want to open it and verify that it works before distributing it for use in your lab.

Procedure

To open a form:

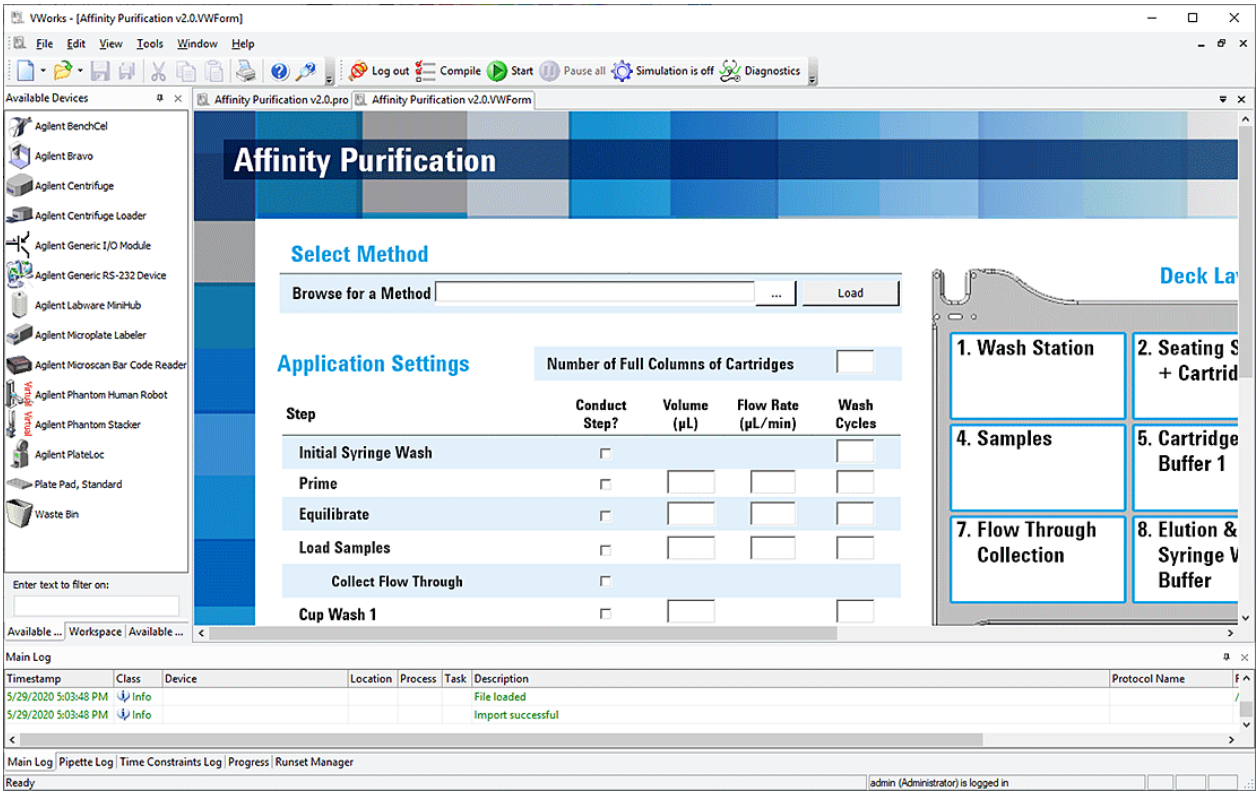
1 Log in to the VWorks software.

2 Click **File > Open >Form.** The Open File dialog box opens.



3 In the **Open File** dialog box, select the form file (*.VWForm), and then click **Open.** The form opens in the VWorks window.

Figure Example of a form open in partial screen mode



Note: A form file is not associated with a protocol or device file until you start a run.

Note: If you need to view or update the inventory database before starting the run, you must first open the related protocol or device file. If you want to display the device file, click the **Workspace** tab in the **Available tasks** area, and then click the device file name.

Related information

For information about...	See...
Creating forms	"Creating forms for protocol operators" on page 177
Opening a protocol	"Opening a protocol" on page 291

Opening a protocol

About this topic

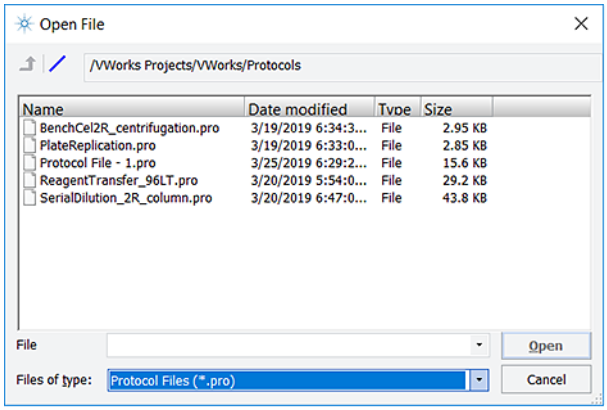
The instructions in this topic assume that the protocol is associated with the correct device file, profiles, teachpoint files, labware definitions, liquid classes, and other relevant files and databases. For details about how these components relate and impact each other, see "Relationship of components in VWorks software" on page 5. For instructions on creating or selecting different device files, profiles, and other relevant files, see "Workflow for creating a basic protocol" on page 22.

Procedure

To open a protocol:

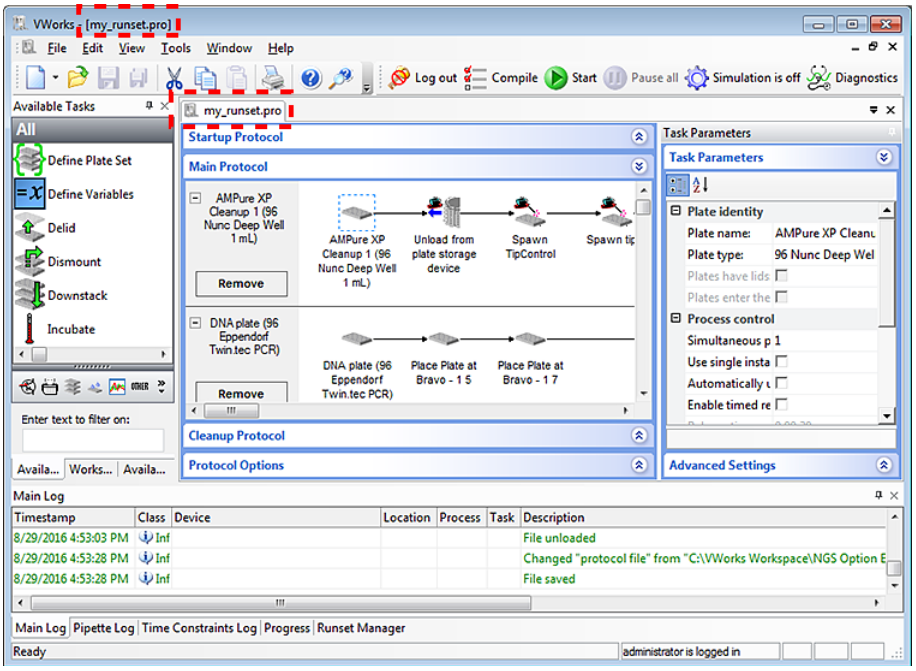
1 Log in to the VWorks software.

2 Click **File > Open Protocol.** The Open File dialog box opens.



3 In the **Open File** dialog box, select the protocol file (*.pro) and then click **Open.**

The protocol information is displayed in the VWorks window. Notice that the name of the protocol appears in the title bar and on the tab in the protocol area.

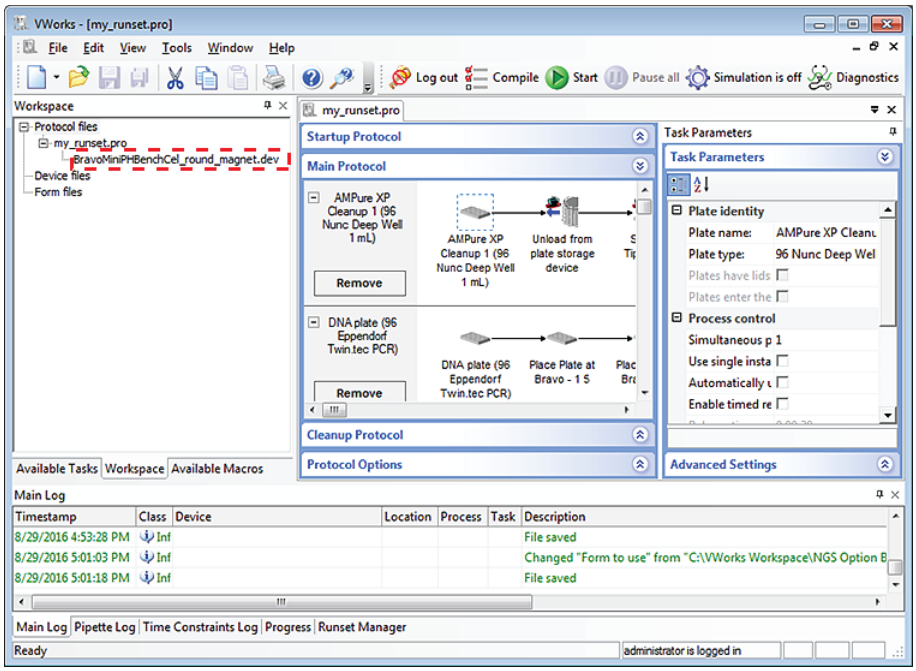


To open multiple protocols:

Repeat the steps in previous procedure to open multiple protocols.

To view a particular protocol, in the Protocol area, click the tab that displays the protocol name.

Note: Every protocol requires a device file to run. When you open a protocol file, the device file associated with it is automatically loaded. However, the device file is not displayed. If you want to display the device file, click the **Workspace** tab in the **Available tasks** area, and then click the device file name.



Related information

For information about...	See...
VWorks software components	"Relationship of components in VWorks software" on page 5
Preparing devices for a run	Device user guide
Creating protocols	"Workflow for creating a basic protocol" on page 22
Setting file directories	"Viewing and setting file directory paths" on page 299
Setting error-handling options	"Setting error-handling options" on page 307
Setting up email notification	"Setting up email notification" on page 309
Running a protocol	"Workflow for running a protocol" on page 288

Disabling and enabling a device in the device file

When to temporarily disable a device

You might want to temporarily disable a device when:

- Running a protocol using a subset of devices
- Running protocols that use different pipette heads or deck setups
- Performing maintenance on a pooled or non-pooled device

The disabled device will be not be available during a protocol run. If the software encounters a task pointing to the disabled device during a run, the protocol will skip the task and continue the run. If the disabled device is part of a pool of devices and the other devices in the pool are available, the task will be executed using the available device.

Note: Up to 10 devices in a single device file can be disabled.

To disable a task without disabling the device, see "Disabling and enabling tasks" on page 723 for more information.

When to enable a device

You might want to enable a device when:

- You are running a protocol that uses a device that was disabled for another protocol
- You have finished performing maintenance on a device

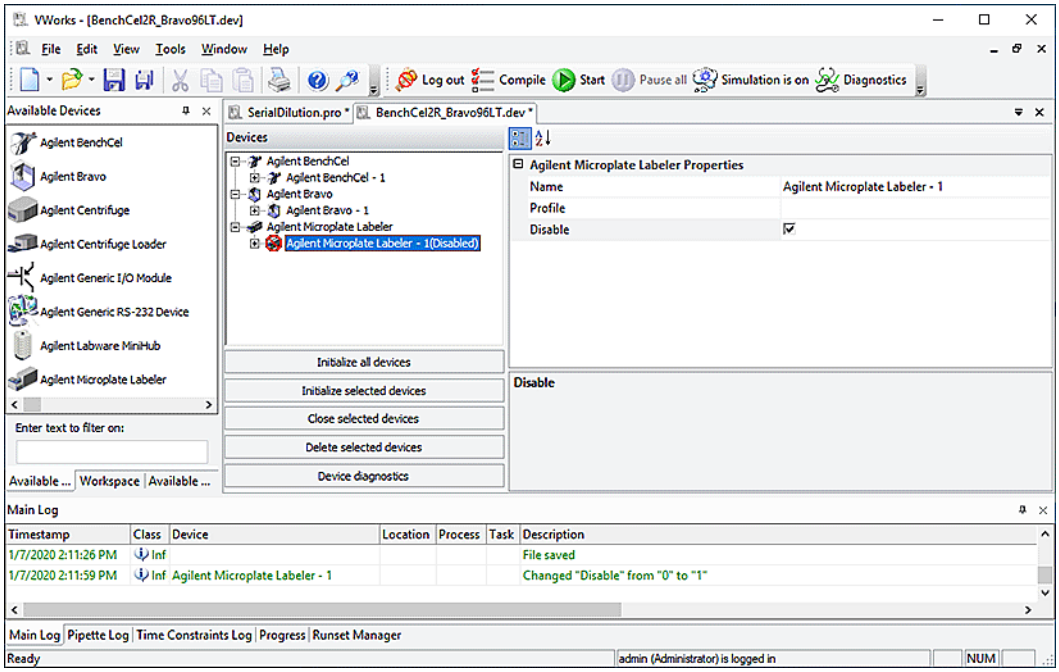
Before you start

Before you enable a device, make sure that you have set the teachpoints for the device. For more information see "Adding devices" on page 26 and the user guide for the specific device.

Procedure

To disable a device in a device file:

1. In the VWorks window, click **File > Open >Device**.
2. In the **Open File** dialog box, select the device file (*.dev), and then click **Open**.
   The device pane appears in the VWorks window and displays the list of devices.
3. Select the device in the **Devices** area and then select the **Disable** check box under the device properties area, or right-click the device and select **Disable device**. In the Device area, appears on the device icon, and (Disabled) is appended to the device name.



To enable a device in a device file:

1. In the VWorks window, click **File > Open >Device**.
2. In the **Open File** dialog box, select the device file (*.dev), and then click **Open**.
   The device pane appears in the VWorks window and displays the list of devices.

3 Clear the **Disable** check box under the device properties area. Alternatively, right-click the device name and select **Enable device**.

In the Devices area, the icon disappears from the device and the name is no longer appended with the (Disabled) status.

Example situations for disabling a device

Running protocols that use different pipette heads or deck setups

For example, you might want to run two protocols that use a Bravo Platform but each one requires a different pipette head. One protocol uses 96-well microplates and the other 384-well microplates. Instead of having two device files, each with a unique Bravo Platform profile for a specific head, you can create one device file in which you add two Bravo Platform devices, each with a unique profile and setup. When running one of the protocols, the Bravo Platform device with the configuration that is not in use is disabled.

1 In the VWorks window, click **File > Open >Device**.

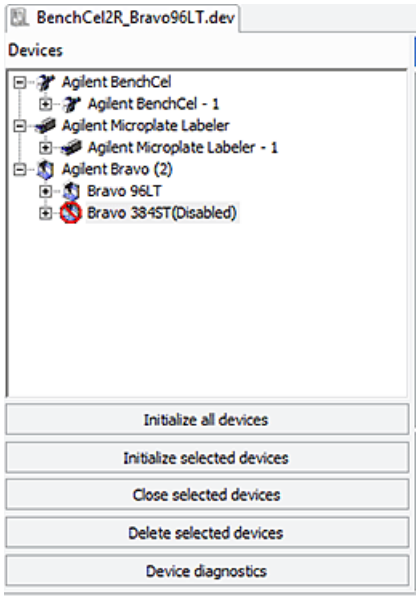
2 In the **Open File** dialog box, select the device file that is associated with one of the protocols that use the Bravo Platform, and then click **Open**.

3 Add a second Bravo Platform and configure it to use the second profile and setup. See "Adding devices" on page 26 for instructions.

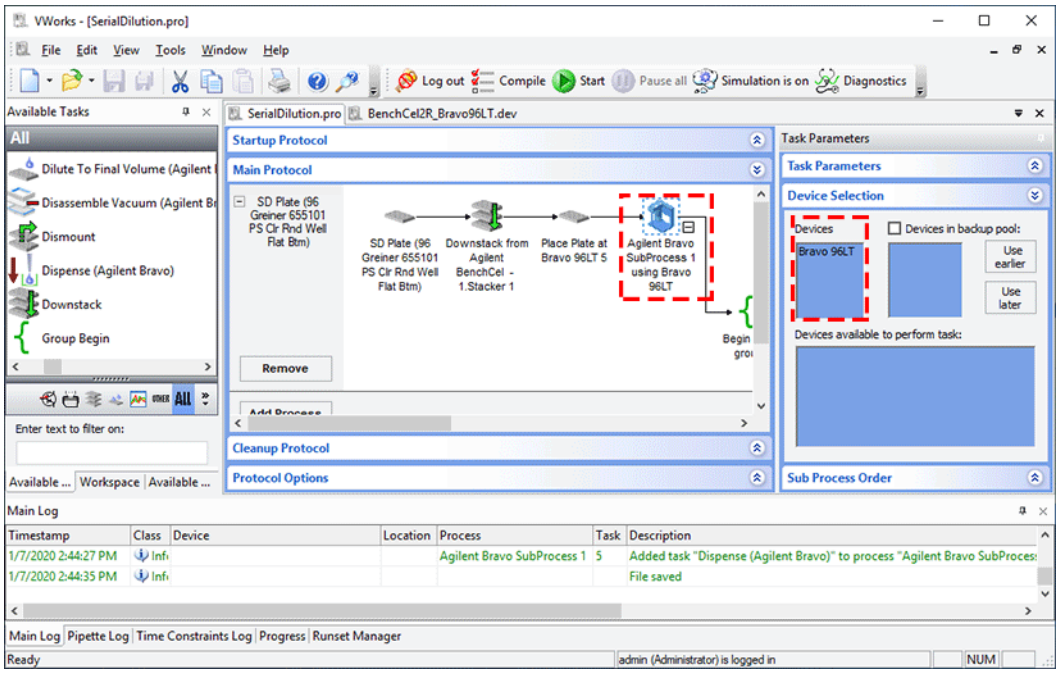
4 Open the protocol you want to run. In this example, it is the one that uses 96-well microplates (Bravo 96LT profile).

5 Disable the Bravo Platform device with the profile (Bravo 384ST) that is not used in the protocol. See "To disable a device in a device file:" on page 294 for instructions.

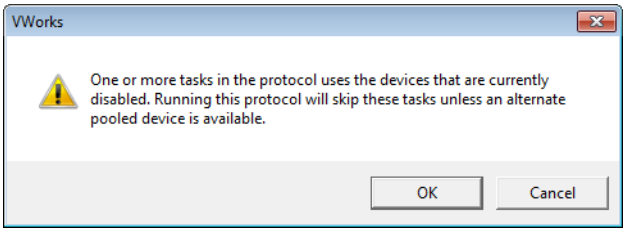
Note: The disabled Bravo device will be removed from the available device list.



6 In the protocol, ensure the Bravo subprocess is associated with the correct Bravo device. See "SubProcess (Agilent Bravo)" on page 450 for instructions.



7 When you are ready to run the protocol, click **Start**. A disabled device message appears and asks you to confirm that you want to skip the disabled tasks.



8 Click **OK** to run the protocol. The Run Configuration Wizard opens. Follow the instructions in the wizard. For details on the wizard, see "Starting the protocol run" on page 312.

Performing maintenance on a non-pooled device

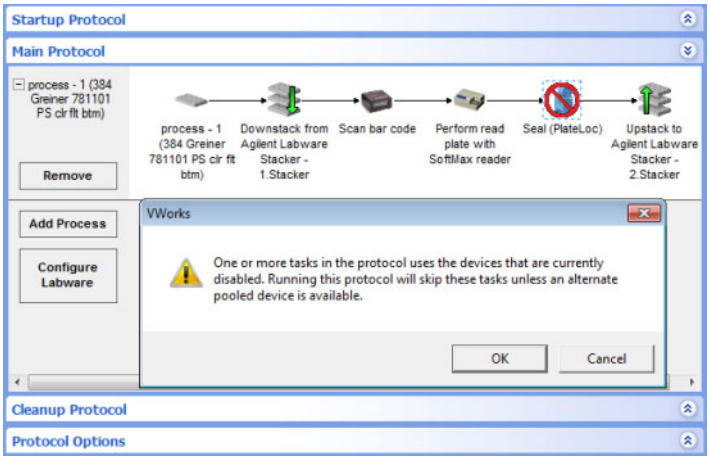
While replacing a roll of seal on a PlateLoc Sealer, you want to continue to perform tasks on the other devices.

1 Open the protocol you want to run.

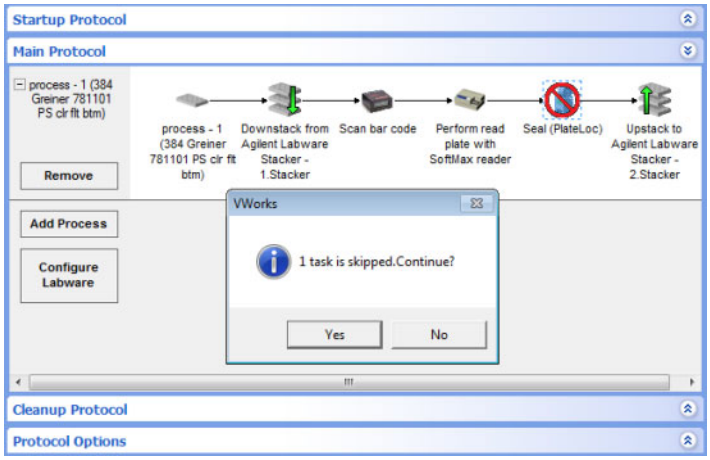
2 Open the device file associated with the protocol.

3 Disable the PlateLoc Sealer associated with the protocol. See "To disable a device in a device file:" on page 294 for instructions.

4 When you are ready to run the protocol, click **Start**. A disabled device message appears and asks you to confirm that you want to skip the disabled tasks.



5 Click **OK** to confirm that the tasks should be skipped. A second message appears and asks you to confirm the number of tasks to be skipped. In the following example, one task is skipped.



6 Click **Yes** to confirm the number. The Run Configuration Wizard opens. Follow the instructions in the wizard. For details on the wizard, see "Starting the protocol run" on page 312. During the run, the tasks associated with the disabled device are skipped. In the preceding example, the Seal task is skipped because the PlateLoc Sealer device is disabled.

Performing maintenance on a pooled device

While performing maintenance on a PlateLoc Sealer that is a member of a pool of PlateLoc Sealers, you want to continue to run a protocol using the other devices.

1 Open the protocol you want to run.

2 Open the device file associated with the protocol.

3 Disable the PlateLoc Sealer you want to service. See "To disable a device in a device file:" on page 294 for instructions.

Note: If the disabled PlateLoc Sealer device was associated with a Seal task, the log contains a warning indicating that because no device is assigned to the Seal task, the task will be skipped during the protocol run.

4 Reassign another PlateLoc Sealer from the device pool to the Seal task and verify that the task parameters are correct.

Note: If another PlateLoc Sealer was placed in the backup pool, the software automatically moves the backup device to the Devices involved in this task area and you will not have to reassign another device.

5 Run the protocol. The software uses the alternate available device.

Note: If you do not reassign another device to the affected task, a message appears when you compile or run the protocol and allows you to skip the task.

Related information

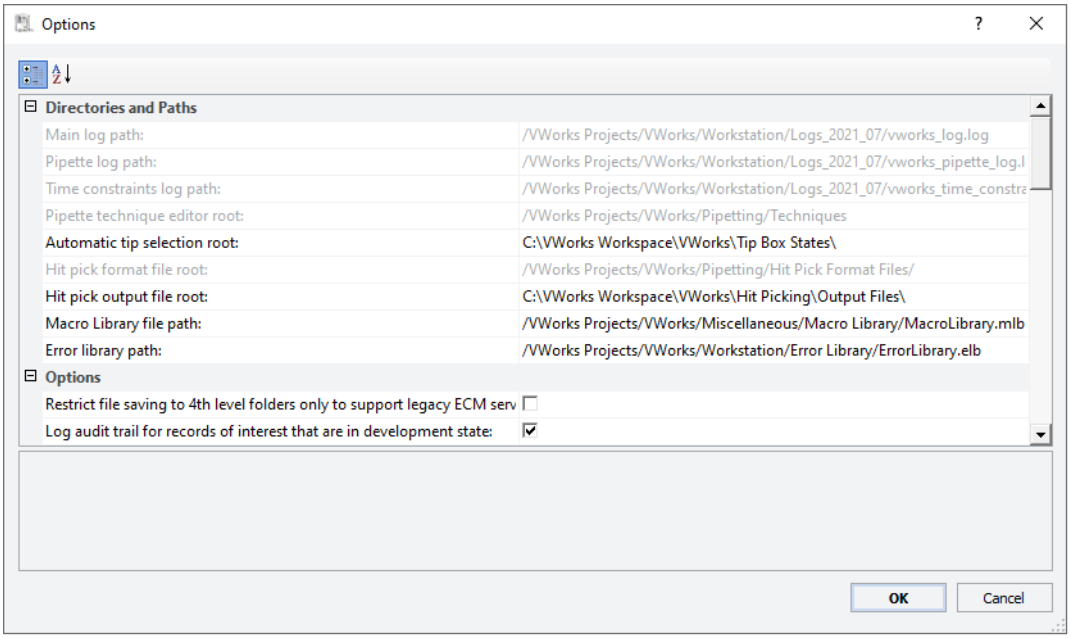
For information about...	See...
VWorks software components	"Relationship of components in VWorks software" on page 5
Adding a device	"Adding devices" on page 26
Preparing devices for a run	Device user guide
Creating protocols	• "Creating a protocol: basic procedure" on page 17 • "Creating a protocol: advanced topics" on page 89
Starting a protocol	"Starting the protocol run" on page 312
Add a SubProcess	"SubProcess (Agilent Bravo)" on page 450
Adding a task	"Adding and deleting tasks" on page 67
Simulating a run	"Simulating the protocol run" on page 80
Running a protocol	"Workflow for running a protocol" on page 288

Viewing and setting file directory paths

Viewing the directory file paths

To view the file paths:

1. Click **Tools > Options**. The Options dialog box opens.
2. Under **Directories and Paths**, notice that the file paths for the directories, except a couple, are relative to the root directory in Shared Services storage, which is set when the software is installed, for example, /VWorks Projects/VWorks/.



About the log files

The software records events that occur during each VWorks session and stores the information in the following logs:

- *Main Log.* Logs all of the actions that occur in the software, including changes to device profiles made in the diagnostics software.
- *Pipette Log.* Logs all pipetting transfer tasks.
- *Time Constraints Log*. Logs all information about time-limited tasks.

In the Options dialog box, the log file paths are unavailable for editing.

Setting file paths for libraries

You can select the path for the following libraries:

- *Error library*. A collection of automated error recovery responses for selected errors. Error library files are stored in the following location:

  /VWorks Projects/VWorks/Workstation/Error Library/

- *Macro library*. A collection of VWorks task macros. Each macro is an abbreviated grouping of task sequences that you can reuse when creating protocols. Macro library files are stored in the following location:

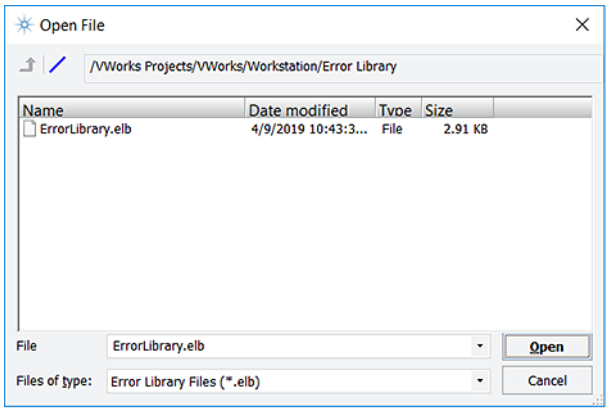
  /VWorks Projects/VWorks/Miscellaneous/Macro Library/

You can use the following procedure to select these library files:

To set the file path for the error library or macro library:

1 In the **Options** dialog box, click the field next to **Error library path** or **Macro library file path** and then click the ··· button that appears. The Open File dialog box opens and displays the directory path for the library.

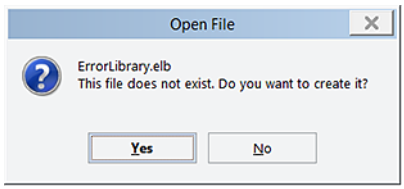
Figure Example of Open File dialog box for error library



2 In the **Open File** dialog box, select the desired library file and click **Open.** The selected library path displays in the Options dialog box.

Note: You can type a new file name to create a new library of the same type in this directory.

If this is the first time to open the library, the message **This file does not exist** appears.



Click **Yes** to create the library, and then refer to one of the following for details:

- Error library, see "Setting up automated error responses" on page 735.

  To enable the error library, select the **Enable error library** check box in the Options dialog box. To disable the error library, clear the check box.

  If you do not want to use an error library, right-click the file path, and then click **Clear field contents**.

- Macro library, see "Using macros to create protocols" on page 265.

  If you do not want to use a macro library, right-click the file path, and then click **Clear field contents**.

3 If you are finished setting options, click **OK** to save the changes and close the Options dialog box.

4 *VWorks Plus only*. In the **Audit Comment** dialog box, select or type a reason for the changes, and then click **OK**.

Related information

For information about...	See...
Log file directory settings	"Viewing logs" on page 714
General options	"Setting general and view options" on page 301
Email option	"Setting up email notification" on page 309
Online option	"Starting the protocol run" on page 312
Setting error-handling options	"Setting error-handling options" on page 307
Running a protocol	"Workflow for running a protocol" on page 288
Setting up the error library	"Setting up automated error responses" on page 735

Setting general and view options

About the global options

You specify settings that apply to all protocols in the Options dialog box. The global settings include:

Options categories	See...
Directories and Paths	"Viewing and setting file directory paths" on page 299
Options (general)	"Setting the general options" on page 302
Email Setup	"Setting up email notification" on page 309
View Options	"Setting the view options" on page 305
Inventory DB Setup	*VWorks Automation Control Setup Guide*
Experiments DB Setup	*VWorks Automation Control Setup Guide*
E-Signatures DB Setup	*VWorks Automation Control Setup Guide*
Error Handling	"Setting error-handling options" on page 307
Watcher Options	"Turning on Watcher" on page 765

Setting the general options

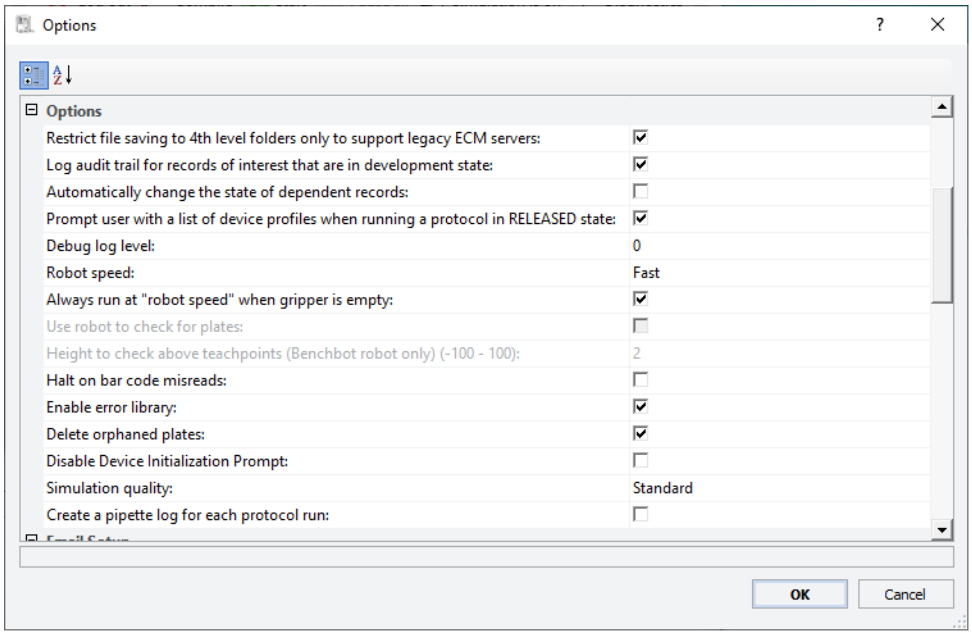
General options allow you to specify the amount of debug information to display in the Main Log, set the robot speed, enable the error library, and other preferences.

You can set the general options during setup or the first time a protocol is run in the software. You do not need to set the general options every time you run a protocol.

CAUTION **The settings in the Options dialog box apply to all protocol runs. Always check the settings before you start a run.**

To set the general options:

1 Select **Tools > Options.** The Options dialog box opens.



2 In the **Options** area, set or select the following options:

Option	Description
Restrict file saving to 4th level folders only to support legacy ECM servers	Select this option to maintain compatibility with legacy Agilent ECM repositories, or clear this option if the compatibility is not required and you want to remove this restriction. For details on the project folder structure, see the *VWorks Automation Control Setup Guide*.
Log audit trail for records of interest that are in development state	*VWorks Plus only*. The option to maintain an audit trail for records that are in the development state. If you select this option, the Audit Comment dialog box will appear anytime you create and save or edit records of interest in the development state. For details on record states, see the *VWorks Automation Control Setup Guide*.

Option	Description
Automatically change the state of dependent records	*VWorks Plus only.* The option to change the state of dependent records when you change the state of a record of interest with dependencies. For details on record states, see the *VWorks Automation Control Setup Guide*.
Prompt user with a list of device profiles when running a protocol in RELEASED state	*VWorks Plus only.* The option to list all the referenced device profiles that are in the validation state when running a protocol that is in the released state, or when running a protocol that references a device file that is in the released state. For details on record states, see the *VWorks Automation Control Setup Guide*.
Debug log level	The amount of debug information presented in the Main Log. This field should be used by Agilent Agilent Technical Support or advanced users for troubleshooting purposes. Valid values are: • *0*. No debug information. This default value is for typical use. • *1–5*. Various levels of plugin information. • *6–100*. Additional debug information. The larger the value, the more information is presented.
Robot speed (not applicable to the Bravo Platform)	The maximum general speed at which a robot, such as the BenchCel Microplate Handler, will move during the run. If you are testing a new protocol or learning to use the system, run the robot at a slow or medium speed to reduce the risk of crashes. In addition to this general robot speed, you can set: • *Robot-handling speed for labware*. Specifies the maximum speed at which the robot can move when handling a specific type of microplate. This parameter is set in the Labware Editor on the Plate Properties tab. If this speed differs from the general robot speed, the robot uses the slower of the two speeds. For more details on the Labware Editor, see the *VWorks Automation Control Setup Guide*. • *Always run at robot speed when gripper is empty*. This option allows the robot to move at a faster speed when no labware is in the gripper and yet retain the ability to move more slowly when carrying labware.

Option	Description
Always run at “robot speed” when gripper is empty	The option to move at the Robot speed setting, above, when no labware is in the robot gripper. For example, if the Robot speed is fast, but the speed for the labware is slow: • *Select the check box (default)*. Allows the robot to move fast when moving to the labware pickup location, while the gripper is still empty. • *Clear the check box*. Results in the robot moving at slow speed both while moving to the pickup location and while picking up and placing the labware. For the maximum throughput, ensure this option is selected.
Use robot to check for plates	Not supported.
Height to check above teachpoints (BenchBot Robot only)	Not supported.
Halt on bar code misreads	The option that requires the software to pause the protocol when a barcode scanned does not match the barcode in the input file. Causes of the misread include missing barcode labels, damaged labels, or wrong labels.
Enable error library	The option to turn on access to the error library. In addition to selecting Enable error library, specify the error library path under Directories and Paths. Also see “Setting error-handling options” on page 307. For details on how to add errors and error handlers to the error library, see “Setting up automated error responses” on page 735.
Delete orphaned plates	The option to clear from memory any labware that were left in storage devices temporarily before the run was aborted. Doing so allows the use of the storage location in the next protocol run. For example, during a run, a labware is placed in a Labware MiniHub slot while it waits for a device. You abort the run. Before you can restart the run, you must physically remove orphaned labware left in the system from the previous run, such as the labware placed temporarily in the Labware MiniHub. However, the software still remembers that the labware is in the Labware MiniHub slot. Selecting this option clears the software memory and permits that location to be used in the new run.
Disable device initialization prompt	The option to automatically try to initialize the devices in a device file each time you open the device file instead of displaying a message that asks if you want to initialize the devices.

Option	Description
Simulation quality	The accuracy of the simulation. Select one of the accuracy levels: • *Standard*. The lowest level of accuracy, because the software does not query the device plugins. Certain physical constraints are not simulated, so this option results in the fastest simulation. • *More accurate green dots*. The next level of accuracy where the simulation might take longer than Standard quality. Similar to the Standard quality option, the software does not query the device plugins for physical constraints. However, the software ensures the green dots in the Protocol area are positioned accurately during the simulation. • *Dots + communicate with plug-ins*. The highest level of accuracy. The software queries the device plugins to ensure every move is physically permissible, resulting in a slower but more accurate simulation.
Create a pipette log for each protocol run	The option to create separate pipette logs for each protocol run.

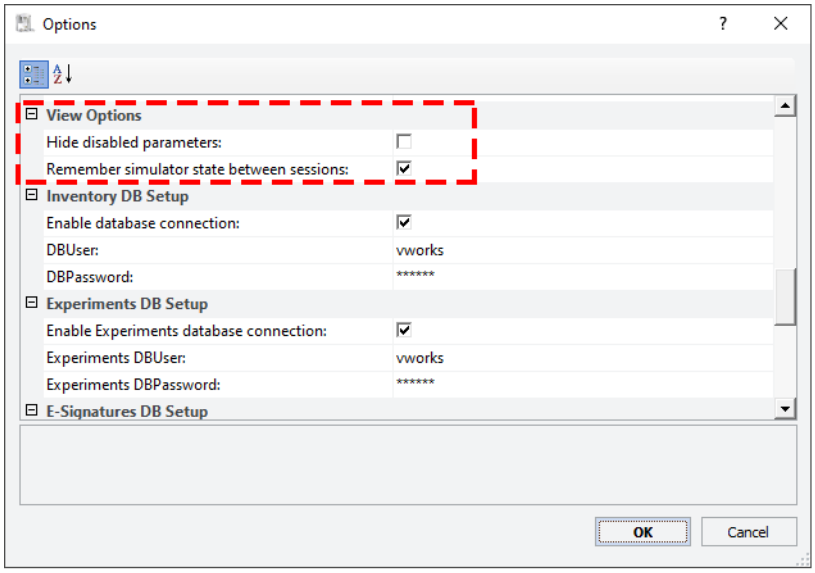
Setting the view options

View options allow you to hide parameters that are not in use and remember the simulation mode when you restart the software.

You set the view options when you first set up the software. You do not need to set the view options every time you run a protocol.

To set the view options:

1 In the **View Options** area, set the following options:



Option	Description
Hide disabled parameters	The option to hide parameters that are not in use: • To hide the unavailable parameters, select the check box. • To show the unavailable parameters as grayed out items, clear the check box. For example, in the Aspirate task, if you select Perform tip touch, additional parameters appear, and you can specify the sides of wells to use for tip touch. If you do not select Perform tip touch, the additional parameters can be hidden or grayed out, depending on whether you selected Hide disabled parameters.
Remember simulator state between sessions	The option to remember the simulation mode (Simulation is on, or Simulation is off) when you restart the software.

2 If you are finished setting options, click **OK** to save the changes and close the Options dialog box.

3 *VWorks Plus only*. In the **Audit Comment** dialog box, select or type a reason for the change, and then click **OK**.

Related information

For information about...	See...
Log file directory settings	"Viewing logs" on page 714
Resolving barcode reader error messages	"Resolving barcode reader error messages" on page 729
Setting error-handling options	"Setting error-handling options" on page 307
Setting online notification options	"Starting the protocol run" on page 312
Running a protocol	"Workflow for running a protocol" on page 288
Setting up the error library	"Setting up automated error responses" on page 735

Setting error-handling options

CAUTION **The settings in the Options dialog box are global settings that apply to all protocol runs. Always check the settings before you start a run.**

About setting automated error responses

VWorks automated error handling includes:

- *Error library*. A collection of automatic responses to specific error messages.

  You select the error library file path and select the Enable error library option in the Options dialog box.

  For details on how to add errors and error handlers to the error library, see "Setting up automated error responses" on page 735.
- *Error-handling options*. You can use the following procedure to set general error-handling options during setup or the first time a protocol is run in the software.

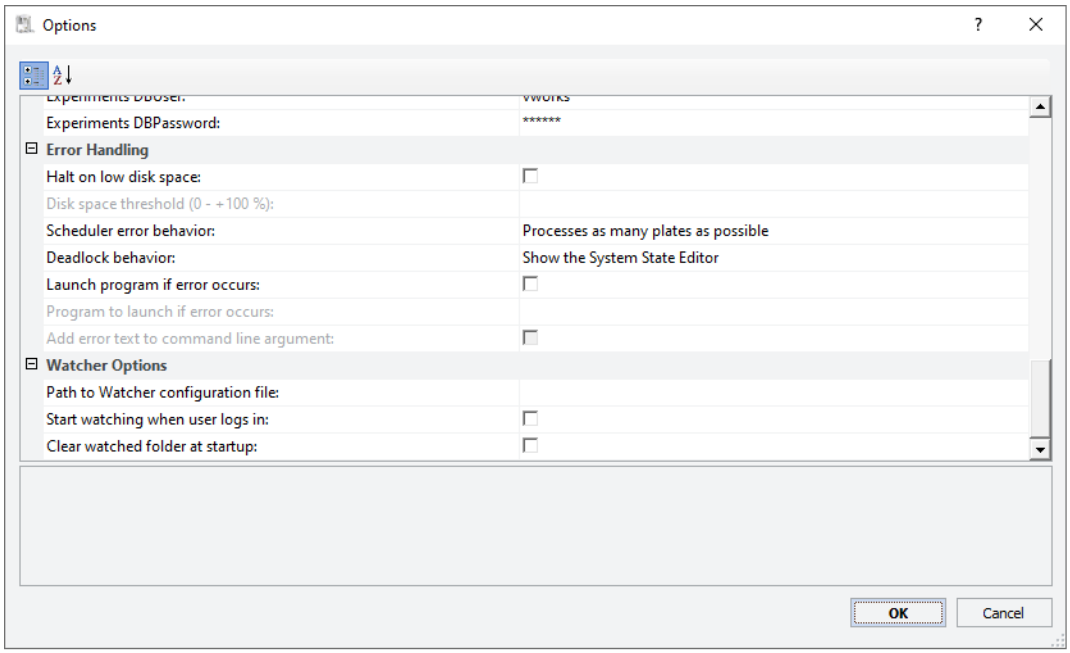
Setting the error-handling options

To set the error-handling options:

1 In the **Options** dialog box, locate the **Error Handling** area.

Note: To open the Options dialog box, click Tools > Options.

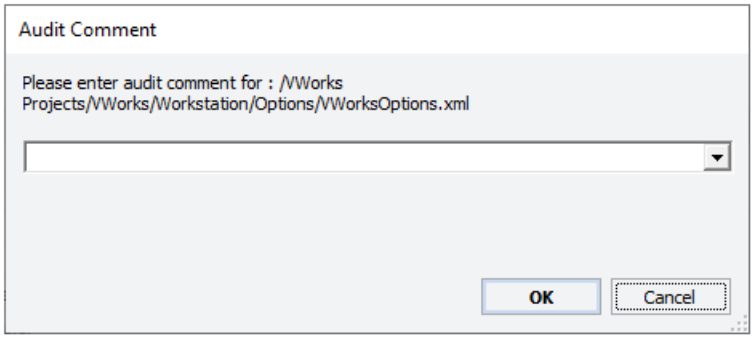
2 Set the following error-handling options:



Option	Description
Halt on low disk space	The option that requires the software to stop scheduling tasks when the computer disk space is low. If you select this option, you must specify the **Disk space threshold.**
Disk space threshold	The percentage of disk space at which the software will halt scheduling tasks.
Scheduler error behavior	The action the software should take if an error occurs during the run. Select one of the following actions: • Process as many plates as possible • Continue processing without starting any new plates • Stop scheduler
Deadlock behavior	The action the software should take if a deadlock occurs during the run. Select one of the following actions: • Abort • Show the System State Editor (pauses the run and displays the System State Editor)
Launch program if error occurs	Starts a specified software application anytime a VWorks error occurs.
Program to launch if error occurs	The file path to the executable file of the application that you want to start when an error occurs. To set the file path, click the field, and then click the ... button that appears. In the Open dialog box, select the file path for the executable file (.exe).
Add error text to command line argument	Passes the text of the error message to the specified application. For example, if this check box is selected and an error occurs containing the text, "Your PlateLoc is out of seal," a command will be generated that says: <filepath>\ABC.exe Your PlateLoc is out of seal If this check box is cleared, the following command would be generated for the same error: <filepath>\ABC.exe

3 If you are finished setting options, click **OK** to save the changes and close the Options dialog box.

4 *VWorks Plus only*. In the **Audit Comment** dialog box, select or type a reason for the changes, and then click **OK**.



Related information

For information about...	See...
Log file directory settings	"Viewing logs" on page 714
System State Editor	"Recovering from deadlocks" on page 730
Automated error responses	"Setting up automated error responses" on page 735
Posting messages about VWorks events, such as errors and deadlocks	"Starting the protocol run" on page 312

Setting up email notification

You can specify that the VWorks software send you an email or text message when a run error occurs. Setting up email notification also enables bug reporting.

Requirements

In addition to specifying the Email Setup options in this procedure, the following prerequisites must be met. Work with your IT organization to meet these requirements.

The computer running the VWorks software must:

- Be connected to a network with internet access.
- Be able to communicate with your outgoing email server (SMTP server). The email server settings must be configured in the Control Panel (OpenLab component).

In the OpenLab Control Panel, go to **Administration > System Configuration > Edit Email Server** to enable communication with your outgoing email server. For instructions on using the Control Panel, see the VWorks configuration and administration guide for your VWorks edition or the Control Panel help topics.

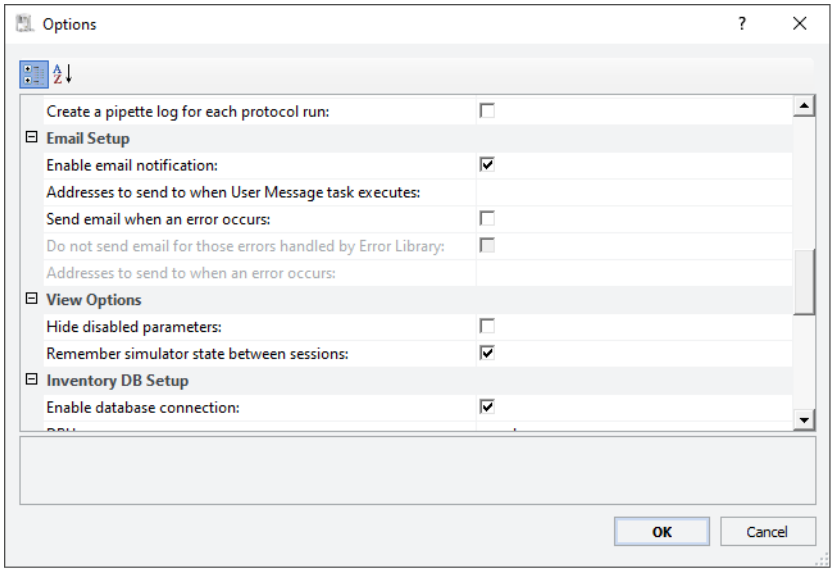
Selecting email setup options

You can enable email notification during setup or the first time a protocol is run in the software. You do not need to select the email notification option every time you run a protocol.

CAUTION **The settings in the Options dialog box apply to all protocol runs. Always check the settings before you start a run.**

To select email setup options:

1 In the **VWorks** window, select **Tools > Options.** The Options dialog box opens.



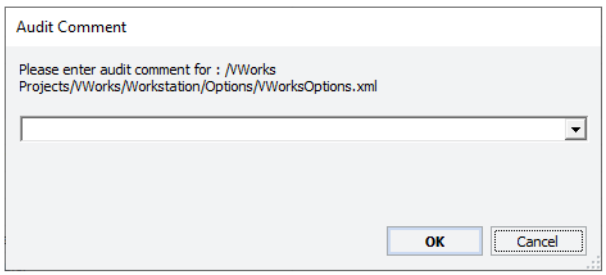
2 Scroll down to the **Email Setup** area, and then specify the following:

Option or parameter	Description
Enable email notification	The option that enables email communication from within the VWorks software. For example, you can report a bug from within the software by selecting Help > Report a Bug. The completed report is emailed to Agilent Technologies. Select the option to enable email communication. Clear the check box to disable email communication.

Option or parameter	Description
Addresses to send to when User Message task executes	The addresses to which the outgoing message is sent when the User Message task runs. Click the browse button and type the email addresses in the Input Text dialog box that appears. Be sure to use semicolons between email addresses. To be notified on your mobile phone with a text message, type the email address for your mobile phone number. The typical format is: number@mobile_carrier_domain.com. Check with your mobile phone service provider. Any charges you might incur and the way messages are delivered and displayed depends on your wireless device and service plan.
Send email when an error occurs	The option to send an email or text message when an error occurs during the run.
Do not send email for those errors handled by Error Library	The option to not send emails if the errors are being handled by the Error Library.
Addresses to send to when an error occurs	The addresses to which the outgoing message is sent when an error occurs. Click the browse button and type the email addresses in the Input Text dialog box that appears. Be sure to use semicolons between email addresses. To be notified on your mobile phone with a text message, type the email address for your mobile phone number. The typical format is: number@mobile_carrier_domain.com. Check with your mobile phone service provider. Any charges you might incur and the way messages are delivered and displayed depends on your wireless device and service plan.

3 If you are finished setting options, click **OK** to save the changes and close the Options dialog box.

4 *VWorks Plus only*. In the **Audit Comment** dialog box, select or type a reason for the change, and then click **OK**.



Related information

For information about...	See...
Reporting problems	"Reporting problems" on page 743
Log file directory settings	"Viewing logs" on page 714
General and view options	"Setting general and view options" on page 301
Error-handling options	"Setting error-handling options" on page 307
Starting a protocol run	"Starting the protocol run" on page 312

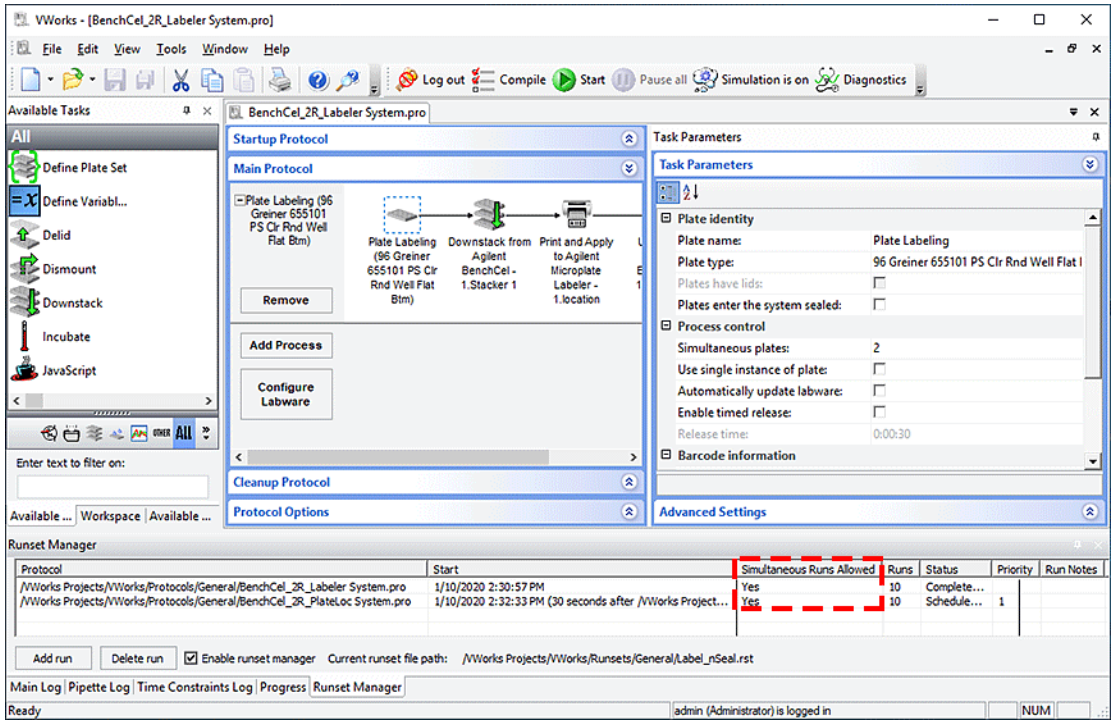
Starting the protocol run

Before you start

The instructions in this topic assume that the protocol is free of errors.

Before you start a run, make sure you review the protocol. To prepare for the run, you should:

- Find out what devices are used in the protocol and prepare them for operation. For example, you might need to load a roll of seal on the PlateLoc Sealer or install a pipette head on a Bravo Platform. See the device user guides for prerun-check and homing instructions.
- Find out which labware are used in the protocol and where they should be positioned before the run starts. For example, you might have to load labware into a stacking device.
- Make sure waste bins are empty and reservoirs are filled.
- Check the protocol for User Message tasks.
- If you are starting a runset, verify whether simultaneous runs should be allowed. You can view the setting for this rule in the Runset Manager tab. To change this protocol rule, see "Specifying protocol rules" on page 46.



About scheduling runs

When you start a run, you have the option of scheduling the protocol to run immediately or at a future date and time. In addition, you can specify that the protocol start while another is still running. Running multiple protocols simultaneously can maximize device use and throughput.

IMPORTANT You can schedule multiple protocols to run simultaneously if they all reference the same device file.

When you schedule multiple runs:

- You can start a run while existing protocols are already running or are scheduled to run.
- Protocols that are running simultaneously can share devices. The priority of device use is specified in the device selection area when you set the task parameters.
- You can manage a set of runs using the Runset Manager. For details, see "Managing runsets" on page 317.

Procedure

You can start a run if you have administrator, technician, or operator privileges.

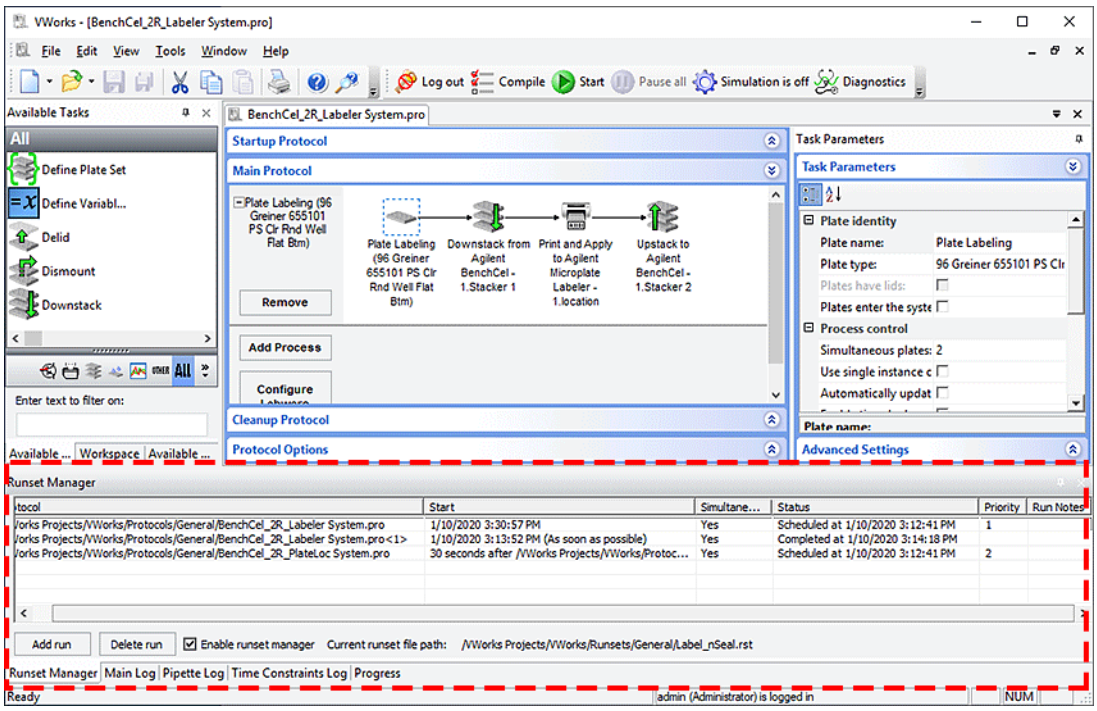
To start a run:

1 In the VWorks window, turn off the simulation mode: On the toolbar, click **Simulation is on**. The button changes to **Simulation is off.**

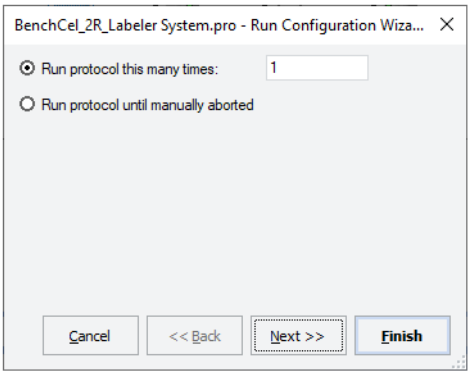


2 Do one of the following:

- Click **Start** (Start) on the toolbar.
- If you have a runset open, view the scheduled start times in the **Runset Manager** tab. To change a protocol start time, double-click the protocol name. For details about runsets, see "Managing runsets" on page 317.



The Run Configuration Wizard dialog box opens.



The Run Configuration Wizard allows you to:

- Specify the number of times to run the protocol.
- Schedule the run to start. You can start the run:
  - As soon as possible
  - On a specific date and at a specific time
  - At a fixed time after a selected protocol starts so that the two protocols are running simultaneously
  - At a fixed time after a selected protocol run ends
- Set the priority of runs (if multiple protocols are scheduled).
- Specify the starting barcode (if any).
- Type notes about the protocol.

CAUTION **If you select the As soon as possible option, the protocol can start to run immediately after you complete the configuration wizard. Before you click Finish, verify that the system is set up and the protocol is ready to run.**

IMPORTANT You must have technician or administrator privileges to set priorities. If you have operator privileges, the runs you start will always have the lowest priorities and the Higher priority and Lower priority buttons will be disabled.

3 Follow the instructions in the wizard and click **Next** or **Finish.**

When you click **Finish,** the run starts if it is scheduled to run as soon as possible.

If this is the first run with the device file, and it is the only protocol running, the software confirms communication with all devices, and then the devices home. (If other protocols are already in progress, the software does not need to establish communication with the devices.)

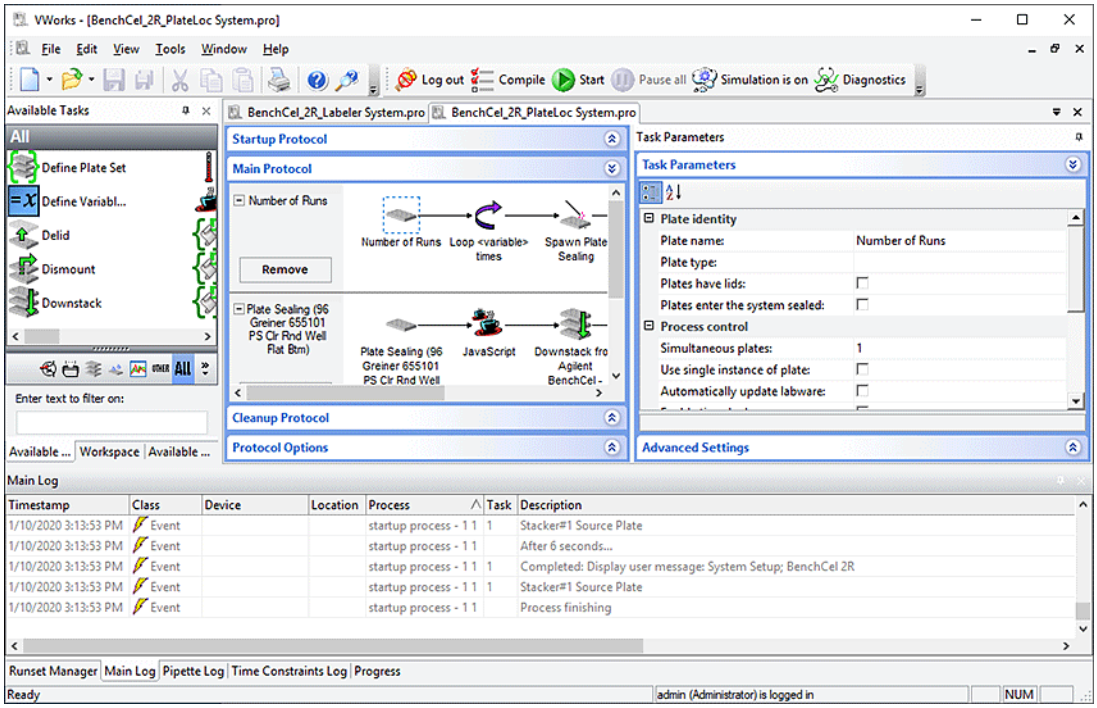
When the run starts

After you start the run, the following occur:

- The **Pause all** (Pause all) button becomes available.
- If the protocol contains tasks pointing to a disabled device, the software displays a warning message and a ⃠ is displayed on the affected tasks. If you click OK, you will get a second message asking you to confirm the number of tasks to be skipped.
- The software checks that the protocol you are starting is using the same device file as the other protocols that are already running. If it uses a different device file, an error message appears.
- The protocol instructions are performed.
- If User Message tasks are included in the protocol, the software prompts you to respond to them as they appear.

  If there are no User Message tasks to remind you to empty liquid waste containers and refill liquid source containers, set your own reminders using lab timers.
- If it is not already listed, the protocol appears in the Runset Manager tab.

- Log messages appear in the Main Log, Pipette Log, and Time Constraints Log tabs. The log messages are recorded in the log file that is stored in the location you specified in the Tools > Options dialog box.



Related information

For information about...	See...
Setting global options	• "Viewing and setting file directory paths" on page 299 • "Setting general and view options" on page 301 • "Setting error-handling options" on page 307 • "Setting up email notification" on page 309
Disabling a device in a device file	"Disabling and enabling a device in the device file" on page 293
Allowing or disallowing simultaneous runs	"Specifying protocol rules" on page 46
Managing runsets	"Managing runsets" on page 317
Setting teachpoints on a device	Device user guide
User Message tasks	• "User Message" on page 682 • "Using simple variables" on page 93
Monitoring the protocol run	"Monitoring the overall run progress" on page 329
Pausing a protocol run	"Pausing the run" on page 337

Managing runsets

Prerequisites

VWorks administrator or technician privileges are required to create a runset.

VWorks Plus only. Runsets are stored as records of interest. All protocols in the runset must be at an equivalent or higher development state to the runset. For example, if the runset is in the validation state, all the protocols in the runset must be in a validation or released state.

About runsets and the Runset Manager

A runset is a collection of protocol runs that can be scheduled in advance to be performed without operator intervention.

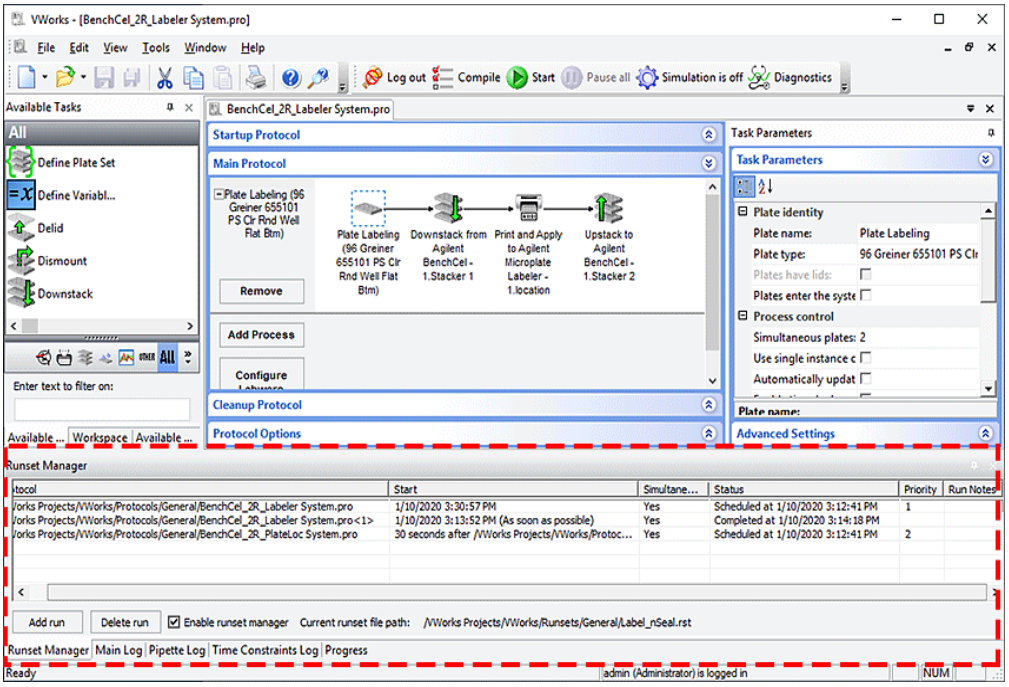
The following table provides an overview of how the Runset Manager works.

Step	Task
1	You add the protocols to a runset and specify the schedule for each protocol, or you can open an existing runset.
2	The Runset Manager determines the most efficient way to run the protocols based on the following: • The specified run start times. • Any specified run priorities. • Whether the protocols share the same device file. • Whether the protocol rules permit simultaneous runs.
3	The Runset Manager opens and compiles each protocol before running it the specified number of times.
4	The Runset Manager repeats step 3 for each protocol in the runset.

The following figure shows the Runset Manager tab, which lists each protocol in the runset on a separate row.

Note: To show or hide the Runset Manager tab, choose **View > Runset Manager**.

Figure A runset displayed in the Runset Manager tab



The Runset Manager tab contains the following columns, which list the runset parameters for each protocol.

Column name	Description
Protocol	The location and name of the protocol file.
Start	The scheduled start time of the protocol.
Simultaneous Runs Allowed	The protocol rule, set in the Protocol Options area, that specifies either: • *Yes (default)*. The Runset Manager can run the protocol while another protocol is running. • *No*. The protocol cannot run simultaneously with another running protocol.
Runs	The number of times the protocol is scheduled to run.

Column name	Description
Status	The status of the protocol: • *Scheduled.* The protocol has a scheduled start time. • *Pending*. The protocol is being prepared to run. • *Running*. The protocol is currently running. • *Completed*. The protocol run has finished. • *Expired*. The protocol was scheduled to start at a time that has already past. • *Paused*. The protocol run is paused. • *Aborting*. The protocol is in the process of being aborted. • *Aborted*. The protocol was aborted. • *No new plates*. The protocol run is not delivering new microplates in the system.
Priority	The schedule priority of the protocol relative to the others in the list, where 1 is the first priority.
Protocol Notes	Any notes about the protocol that were entered in the Run Configuration Wizard.

Understanding how the run sequence is determined

The Runset Manager determines the run sequence of the protocols based on:

- *Scheduled start times*
  - As soon as possible (default)
  - On a specific date and at a specific time
  - Dependencies on other protocols:
    - At a fixed time after a selected protocol starts
    - At a fixed time after a selected protocol run ends

    The Runset Manager handles any protocols with such dependencies as a unit.
- *Whether the protocols have run priorities*. If the protocols in a runset have conflicting run times, the Runset Manager uses the priority settings to resolve which protocol to schedule before the others.
- *Whether simultaneous runs are allowed.* The Runset Manager may reschedule protocol start times if the protocol rules allow simultaneous runs. By default, protocols allow simultaneous runs. You can change this rule for each protocol in the Protocol Options area. See "Specifying protocol rules" on page 46.

The following scenarios provide a few examples.

Runset scenario 1. Start times with dependencies on other protocols

The Runset Manager processes the protocols in the set that have start time dependencies on one another as a unit. For example, assume that a runset includes the following four protocols, each of which can be completed in 10 minutes. The Runset Manager would change the run sequence as follows:

Run sequence: A, C, D, B

Protocol	Device file	Start time
A	1	1:00 pm
B	2	1:15 pm
C	1	30 minutes after protocol A ends
D	1	60 min. after protocol A ends

Even though no other protocols are running at 1:15 pm, the Runset Manager will reschedule the protocol B start time because of the start time dependencies of protocols C and D on protocol A.

Runset scenario 2. Start times with specific dates and times

If you change the start time of the protocols from scenario 1 to the specific start times in the following table, the run sequence would be as follows:

Run sequence: A, B, C, D

Protocol	Device file	Start time
A	1	1:00 pm
B	2	1:15 pm
C	1	1:30 pm
D	1	1:45 pm

Each protocol's start time is independent of the others in the set, so the Runset Manager processes each protocol individually. Although protocol B uses a different device file than the others, no other protocol is running at 1:15 pm, so the sequence is unchanged.

Scenario 3. As-soon-as-possible start times

If you change the start times of the protocols from scenario 1 to the as-soon-as-possible (ASAP) option, the run sequence would be as follows:

Run sequence: A, C, D, B

Protocol	Device file	Start time
A	1	As soon as possible
B	2	As soon as possible
C	1	As soon as possible
D	1	As soon as possible

Because protocol B does not use the same device file, the Runset Manager postpones the start until after completing the runs that share a common device file.

Additionally, if a protocol cannot be started for some reason, the Runset Manager can dynamically adjust the timing of the protocols that have as-soon-as-possible start times. For example, if protocol A cannot be started, the Runset Manager could change the protocol A start to depend on the protocol B end. If protocol A still fails to start after protocol B ends, the Runset Manager could change the protocol A start to depend on the protocol C end. If additional protocols were in the set, the start time could continue to be adjusted in this way until protocol A could be started successfully.

Simultaneous runs

If simultaneous runs were allowed in the previous runset scenarios:

- *Scenarios 1 and 2*. The run sequences would not change because of the specified start times and dependencies.
- *Scenario 3*. The run sequence could change to ACD,B. The protocols that share device file 1 would run simultaneously. After the ACD runs ended, the Runset Manager would initialize the device file 2 devices and start protocol B.

Creating a runset

To create a runset:

1 Choose **File > New > Runset**.

Note: If the Runset Manager tab is not visible, choose **View > Runset Manager**.

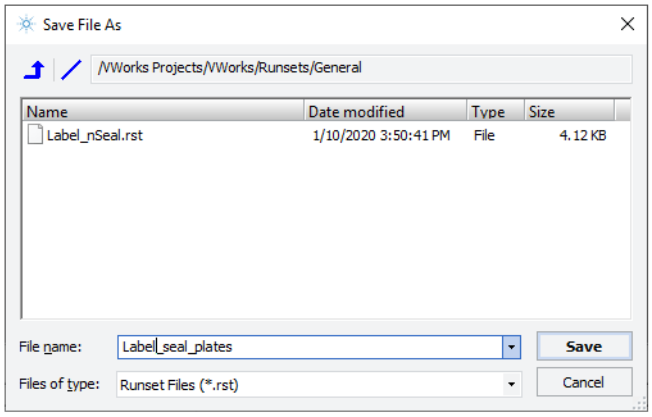
2 Add protocols to the runset using one of the methods in "Adding protocols to a runset" on page 323.

IMPORTANT

Simultaneously running protocols must specify the same static or configured labware.

3 Select **File > Save Runset.**

4 In the **Save File As** dialog box, specify the file name for the .rst file, and then click **Save**.

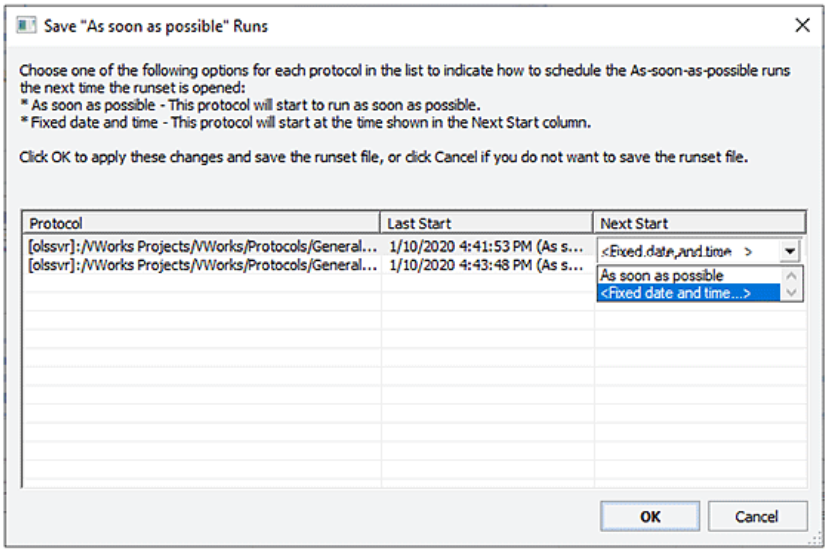


VWorks Plus only. If audit trails are being logged for records in the development state, the **Audit Comment** dialog box opens. Select or type a reason for creating the runset, and then click **OK**.

If any of the protocols in the runset have the **As soon as possible** designation, proceed to step 5. Otherwise, the software saves the runset in Shared Services storage, for example,

/VWorks Projects/VWorks/Runsets/General

5 In the **Save "As soon as possible" Runs** dialog box, verify that all the protocols with the **As soon as possible** setting can be run as soon as possible upon reopening the runset.



To change the **Next Start** setting, click the field in the column, and select **Fixed date and time**. Upon reopening the runset, the operator will be prompted to provide a new start time.

Click **OK** to save the runset.

Alternatively, you can simply add protocols to the runset using one of the methods in "Adding protocols to a runset" on page 323, and then choose **File > Save Runset As**.

If you want to export a runset for use on another computer, see "Exporting and importing protocols and associated records" on page 703.

Adding protocols to a runset

CAUTION **To avoid potential hardware crashes, verify whether each protocol that you add can run simultaneously with the other protocols in the runset. To change the protocol rules, see "Specifying protocol rules" on page 46.**

CAUTION **If you select the As soon as possible option, the protocol can start to run immediately after you complete the configuration wizard. Before you click Finish, verify that the system is set up and the protocol is ready to run.**

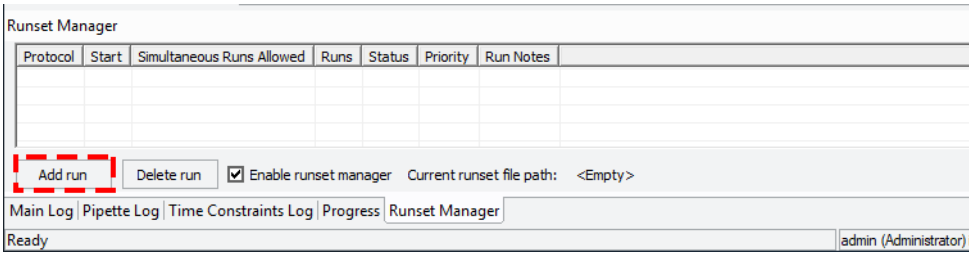
IMPORTANT Simultaneously running protocols must specify the same static or configured labware.

You can add a protocol to a runset in the following ways.

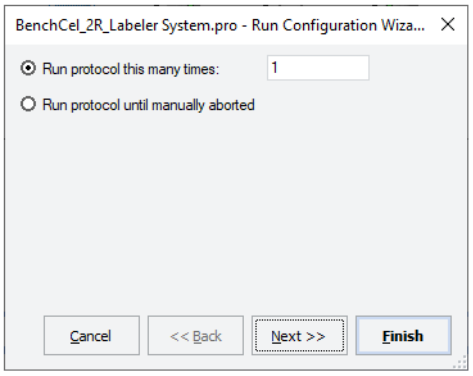
- Use the Add run button in the Runset Manager tab.
- Drag the protocol file name from the Workspace tab to the Runset Manager tab.
- Start a protocol run. See "Starting the protocol run" on page 312.

To add protocols using the Add run button:

1 In the **Runset Manager** tab, click **Add run**.



2 When the **Open a protocol file** dialog box appears, select the protocol (.pro), and then click **Open**. The Run Configuration Wizard appears.



3 In the Run Configuration Wizard:

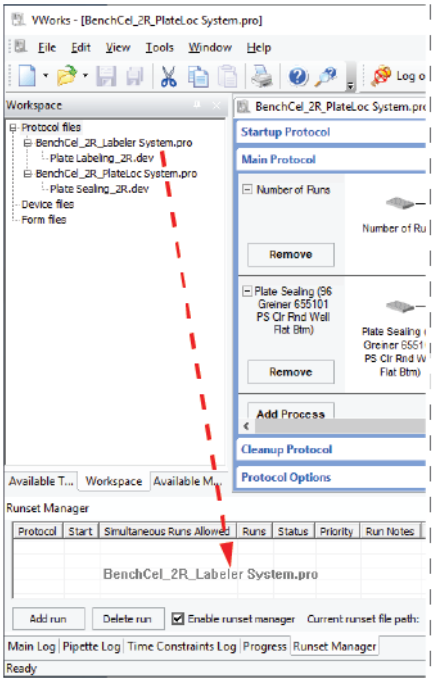
- Specify the number of times to run the protocol.
- Schedule the run start times.
- Set the priority of runs (if multiple protocols are scheduled).
- Specify the starting barcode (if any).

- *Optional*. Type notes about the protocol.

4 Repeat steps 1 to 3 for each protocol you want to add.

To add protocols using the drag-and-drop feature:

1 Drag the file name of the protocol from the **Workspace** tab to the **Runset Manager** tab.



Note: To add more protocols to the Workspace tab, see "Opening a protocol" on page 291.

2 In the Run Configuration Wizard:

- Specify the number of times to run the protocol.
- Schedule the run start time.
- Set the priority of runs (if multiple protocols are scheduled).
- Specify the starting barcode (if any).
- *Optional*. Type notes about the protocol.

3 Repeat steps 1 and 2 for each protocol you want to add.

Opening runsets

IMPORTANT

Verify whether the protocols in the runset can run simultaneously with other protocols (default). To change the protocol rules, see "Specifying protocol rules" on page 46.

If you want to import a runset, see "Exporting and importing protocols and associated records" on page 703.

To open a runset:

1 *Optional*. To simulate the protocols in the runset before starting the actual runs, turn on the simulation mode. To turn on simulation, click **Simulation is off** on the toolbar. The button changes to **Simulation is on.**



2 Select **File > Open.** The Open dialog box appears.

3 Select the .rst file, and then click **Open.**

IMPORTANT

If a protocol in the runset is scheduled to run as soon as possible, a message warns you that the run will start immediately when you open the runset. Make sure that the system is ready for the run to start before you continue.

Filtering the list of protocols displayed

You can apply a filter to display a subset of the protocols in the runset.

To filter the list of runs displayed in the Runset Manager tab:

1 Right-click anywhere in the **Runset Manager** tab.

2 In the shortcut menu that appears, select the desired filter:

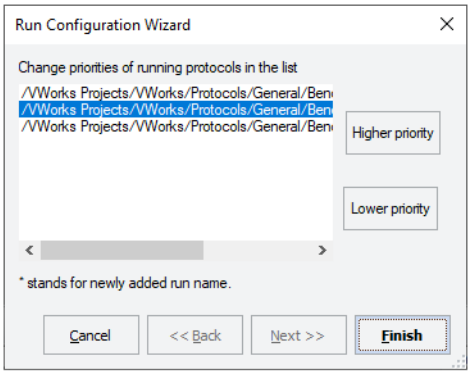
Filter command	Description
Show all	Displays all protocols in the runset.
Filter by row	Displays the protocols that meet the criteria you select. For example, you can display all protocols that are run 10 times.
Filter by column	Displays the protocols that meet the criteria you select.
Use last filter	Displays the protocols that meet the last filter criteria you selected.

Changing run priorities

To change the priority of the protocol runs:

1 In the **Runset Manager** tab, right-click anywhere in the protocol table.

2 In the shortcut menu that appears, select **Adjust run priority.** The Run Configuration Wizard dialog box appears.



3 Select the protocol whose priority you want to change, and then click **Higher Priority** to move it up the list, or click **Lower Priority** to move it down the list.

4 Click **Finish.** The Priority column shows the change in priority.

Aborting a protocol in a runset

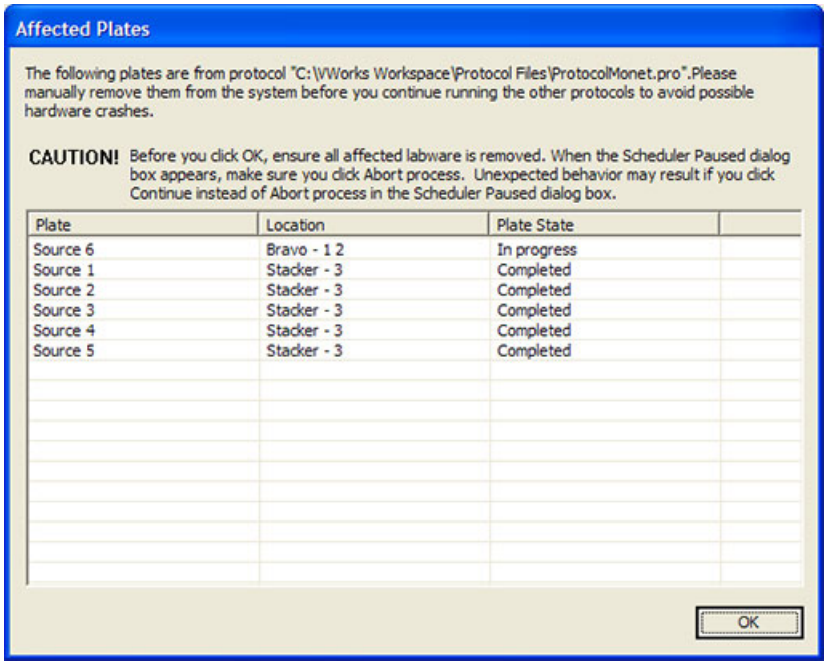
If you abort a protocol in a runset, the software also aborts any other protocols that have time dependencies associated with the aborted protocol.

To abort a protocol in a runset:

1 In the **Runset Manager** tab, right-click the protocol name, and then choose **Abort this run** from the shortcut menu.

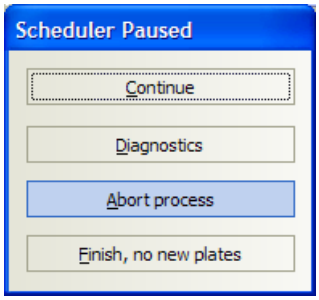
- *If the protocol is scheduled but has not yet started*. The software aborts the protocol and any other protocols that have dependencies on the aborted protocol. The Status column in the Runset Manager tab shows the time at which the protocols were aborted.
- *If the protocol has already started*. The software aborts the protocol. The Status column in the Runset Manager tab shows the time at which the protocol was aborted.
- *If you abort a running protocol and subsequent runs are scheduled*. The Affected Plates dialog box appears and displays the status of all the labware in the selected protocol.

CAUTION **Before you click OK, proceed to step 2. Manually remove all the labware associated with the aborted protocol to ensure that you avoid a potential hardware crash.**

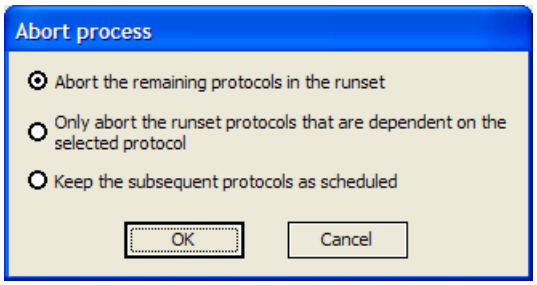


2 Manually remove the physical labware, which is associated with the aborted protocol, from the system.

3 In the **Affected Plates** dialog box, click **OK**. The Scheduler Paused dialog box appears, and the Status column in the Runset Manager tab indicates that the protocol is aborting.



4 In the **Scheduler Paused** dialog box, click **Abort process**. The Abort Process dialog box appears.



5 In the **Abort process** dialog box, choose one of the following options, and then click **OK**:

- Abort the remaining protocols in the runset
- Only abort the runset protocols that are dependent on the selected protocol

- Keep the subsequent protocols as scheduled

The Runset Manager updates the Status column.

Note: For details on the other options in the Scheduler Paused dialog box, see "Pausing the run" on page 337.

6 Verify that the remaining protocols in the runset have appropriate start times.

Rescheduling a protocol in a runset

To reschedule a protocol in a runset:

1 In the **Runset Manager** tab, right-click the protocol name, and choose **Adjust run start time and dependencies** from the shortcut menu.

2 In the **Run Configuration Wizard**, reset the protocol start time. The software automatically resets any other protocols that have time dependencies associated with the rescheduled protocol.

Deleting protocols in a runset

You can use the following procedure to delete a protocol that is not currently running. To abort a protocol that is in progress, see "Aborting a protocol in a runset" on page 326.

To delete a protocol from a runset:

1 In the **Runset Manager** tab, select the protocol.

2 Click **Delete run.** If no other protocols have dependencies on the selected protocol, the software removes the protocol from the list.

If other protocols in the runset have start time dependencies on the selected protocol, a message appears and warns you that continuing will abort the dependent protocols. After the protocol is deleted, adjust the start times of the remaining protocols.

Related information

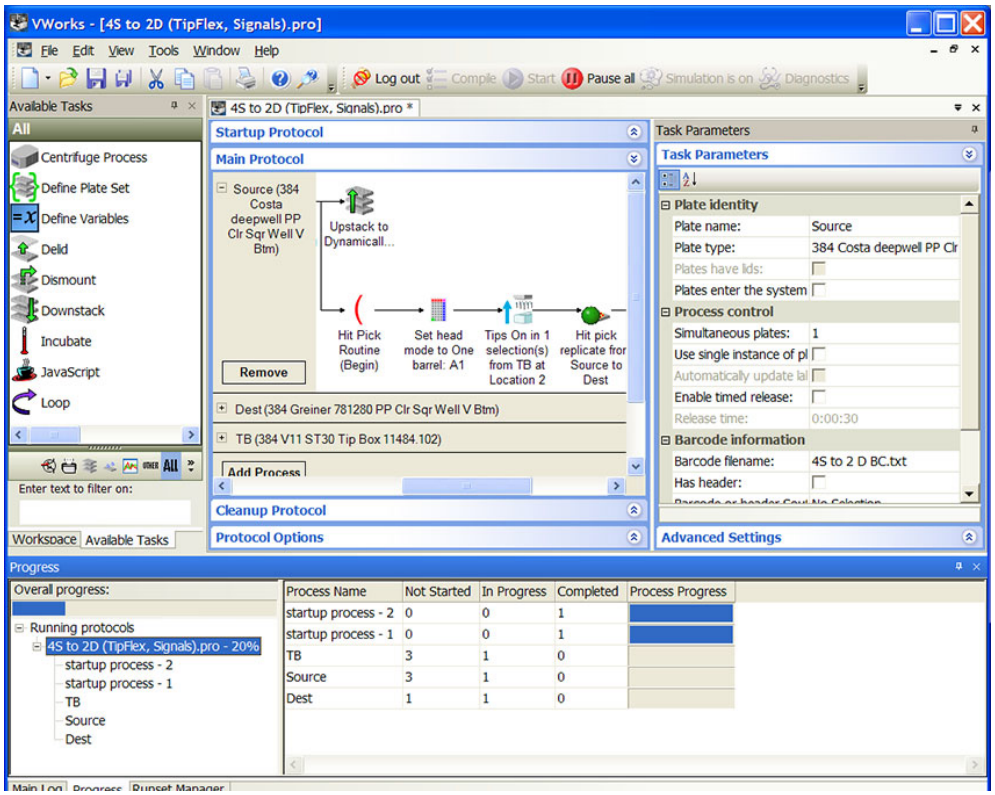
For information about...	See...
Allowing or disallowing simultaneous runs	"Specifying protocol rules" on page 46
Starting a protocol	"Starting the protocol run" on page 312
Monitoring runs	"Monitoring the overall run progress" on page 329
Pausing runs	"Pausing the run" on page 337

Monitoring the overall run progress

Procedure

To check the overall run progress:

In the VWorks window, select **View > Progress.** The Progress tab appears at the bottom of the window and displays the following:



Area or table	Description
Overall progress	Displays a progress bar to indicate how far along the system is in all running protocols.
Running protocols	Contains the list of protocols that are currently running. You can select a protocol name to display the progress of all processes in the protocol, or you can select a specific process and display its progress.
Display	Based on your selection in the Overall progress area, shows an aspect of the overall run progress.

Related information

For information about...	See...
Tracking run progress of instances, processes, or devices	"Tracking the run progress of instances or devices" on page 330
Pausing the run	"Pausing the run" on page 337
Stopping the run	"Stopping the run" on page 341

Tracking the run progress of instances or devices

About this topic

This topic explains the following:

- "Displaying the run progress in the Gantt Chart format" on page 330
- "Displaying the progress by plate instance" on page 332
- "Displaying the progress by device" on page 333
- "Zooming the display" on page 333
- "Filtering the displayed information" on page 334
- "Opening the System State Editor" on page 335
- "Exporting Gantt charts" on page 335

Displaying the run progress in the Gantt Chart format

You can display a Gantt chart for one or more protocols while they are running, or for one or more open protocols have been run in the current VWorks session since they were opened.

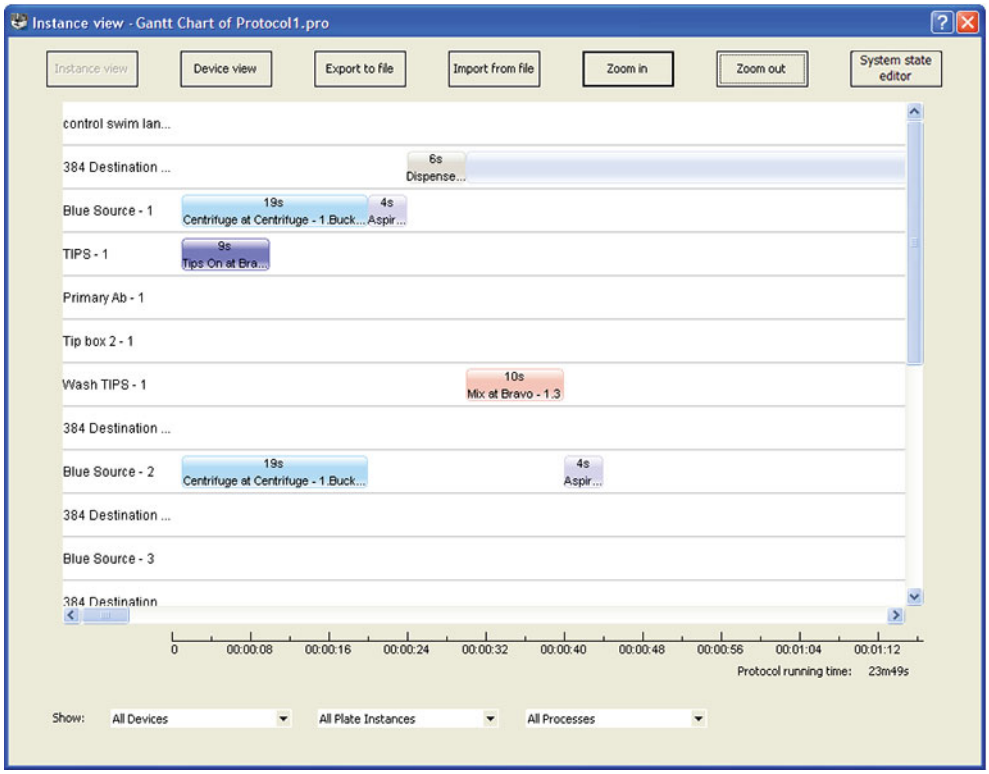
While a protocol is running, you can visually monitor the progress in real-time in the Gantt Chart dialog box. In addition, you have the option of displaying the progress by instance or device.

Run information displayed in the Gantt Chart format enables you to monitor resource usage during the run. Too much spacing (time) between tasks might indicate poor device usage or potential bottlenecks. After you examine the run information, you can:

- Determine the causes of bottlenecks and remove them.
- Add devices or try to run multiple protocols simultaneously to improve performance and throughput.

To visually track the progress of labware instances or devices:

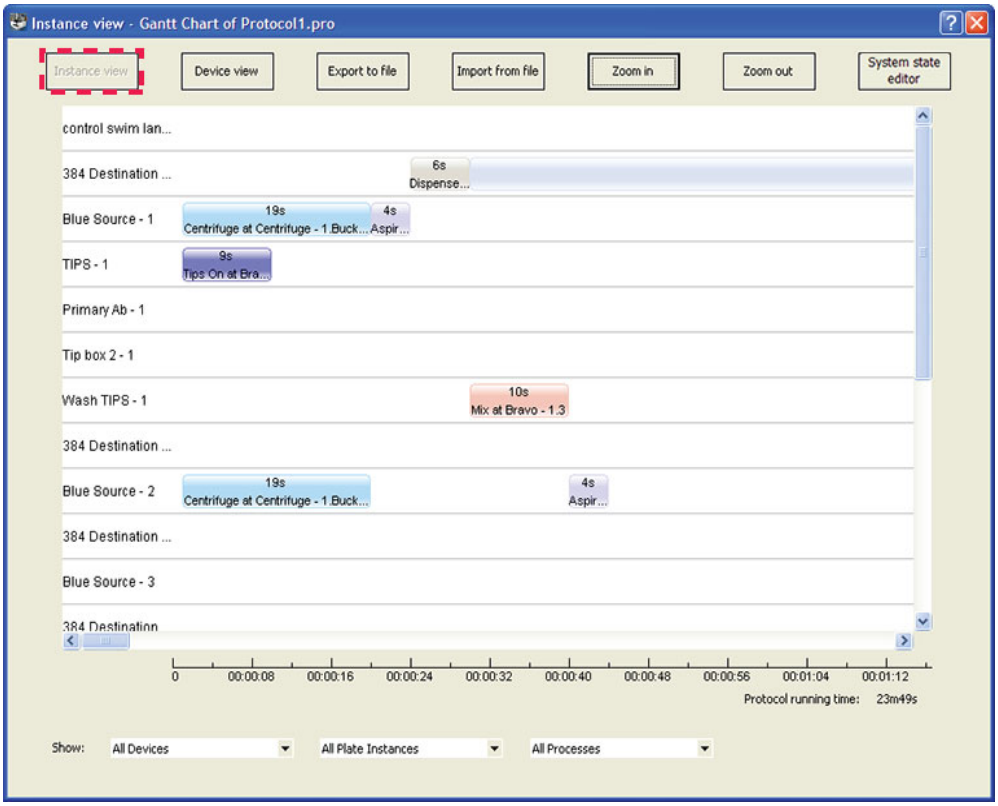
In the VWorks window, select **Tools > Gantt Charts** while a run is in progress, after a run is finished, or when a run has a deadlock error. The Gantt Chart dialog box opens.



Displaying the progress by plate instance

To display the progress by individual plate instances:

In the **Gantt Chart** dialog box, click **Instance view.**



The Instance View presents a graph of the process plate instances (vertical axis) as a function of time (horizontal axis). Each row represents the timeline of a process plate instance. During a run, a row displays the tasks that are performed on a plate instance. The length of the task block represents the duration of the task. The actual duration of each task is displayed on the task block.

Note: A task that does not involve a process plate is not displayed.

If an error occurred during the run, you can use the Instance view to determine the plate instance at which the error occurred. If a deadlock occurred or if the run was aborted, a red X symbol appears on the task at which it occurred.

From within the Instance view, you can double-click a task to edit its parameters.

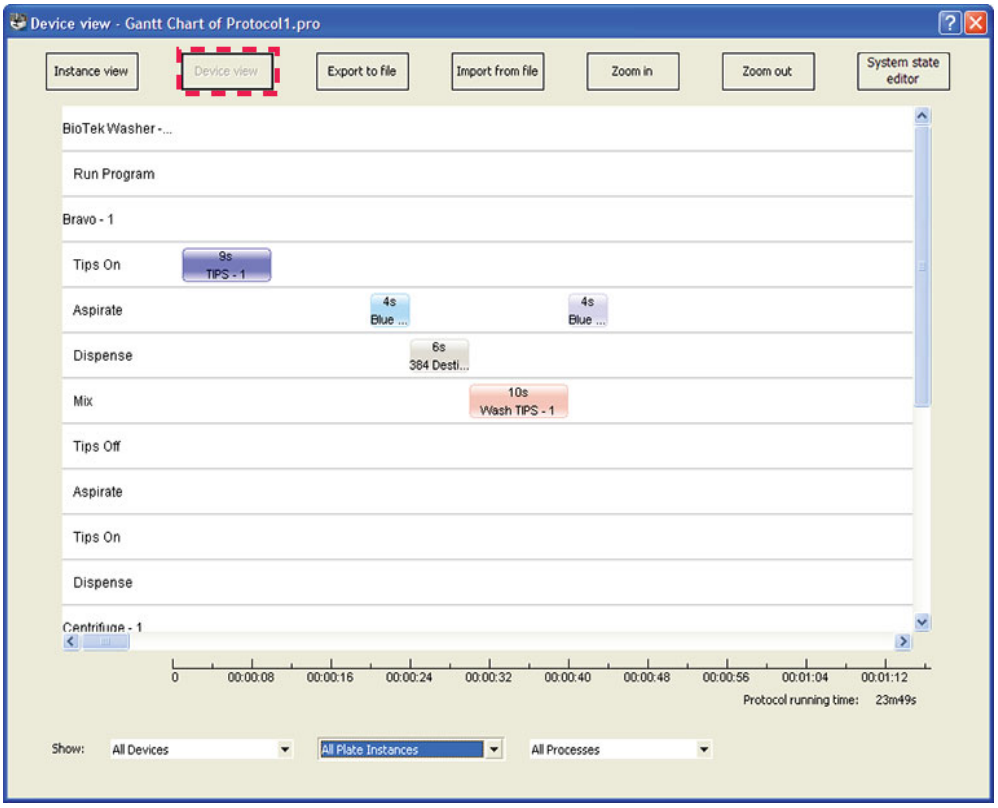
To edit a selected task parameter within the Instance view:

Right-click a task in the graph, and then select **Edit Parameter.** The software minimizes the Gantt Chart dialog box and displays the task within the protocol so that you can edit the parameters.

Displaying the progress by device

To display the progress by device:

In the **Gantt Chart** dialog box, click **Device view.**



The Device View presents a graph of devices (vertical axis) as a function of time (horizontal axis). A row represents the timeline of a device. During a run, each row expands and lists the tasks that the device is performing. The process plate on which the task is performed is shown on the task timeline. The length of the process-plate block represents the duration of the task on the plate.

To optimize the run, look for wide spacing between tasks and determine whether adding devices to the run can improve the throughput. Ideally, the spacing between the tasks should be kept to a minimum.

Zooming the display

You can zoom in or zoom out to change the horizontal (time) scale of the graphs. Each time you zoom in, you are reducing the time increments displayed. Each time you zoom out, you are increasing the time increments displayed.

For example, if the current scale is in increments of 4 seconds, zooming in reduces the increments to 2 seconds. Zooming out increases the increments to 8 seconds.

Note: Zooming has no affect on the scale of the vertical-axis.

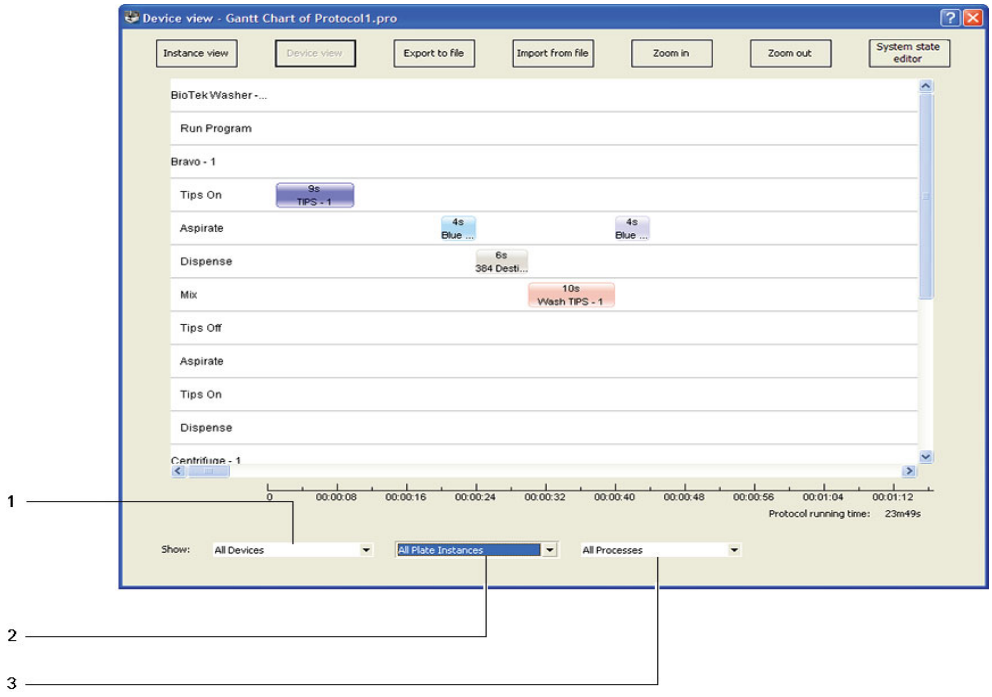
To zoom in or zoom out of the current view:

Click **Zoom in** or **Zoom out.**

Filtering the displayed information

While in the instance or device view, you can filter the information displayed to focus on areas of interest. For example, you can choose to display the information from a single device instead of all devices in the run.

The filter selections are at the bottom of the Gantt Chart dialog box.



To filter the information displayed in the Gantt Chart dialog box:

1 Select one of the following from the Device list:

Selection	Description
All devices	Displays all devices used in the protocol.
Selected devices	Allows you to specify which devices to display in the dialog box.

2 Select one of the following from the Plate Instances list:

Selection	Description
All plate instances	Displays all plate instances.
Completed plate instances	Displays only plate instances that have finished processing.
Plate instances in progress	Displays only plate instances that are currently in progress.

3 Select one of the following from the Process Plate list:

Selection	Description
All processes	Displays all plate processes.

Selection	Description
Completed processes	Displays only plate processes that are finished.
Processes in progress	Displays only plate processes that are currently in progress.
Selected processes	Displays only the plate processes that you have selected.

Opening the System State Editor

You can open the System State Editor to recover from a deadlock error.

To open the System State Editor:

In the **Gantt Chart** dialog box, click **System State Editor.** The System State Editor dialog box opens.

For information about deadlock recovery and the System State Editor, see "Recovering from deadlocks" on page 730.

Note: If you want the System State Editor dialog box to open automatically whenever a deadlock occurs, choose **Tools > Options**. Under **Error Handling** in the Options dialog box, choose **Deadlock behavior > Show the System State Editor**. For more information, see "Recovering from deadlocks" on page 730.

Exporting Gantt charts

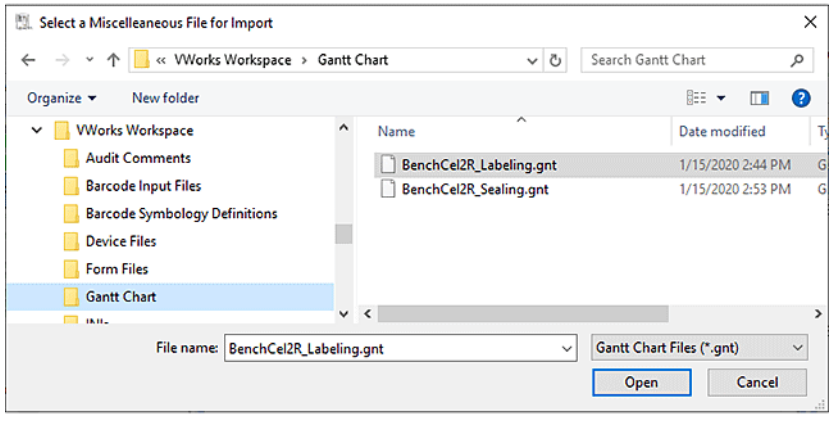
To export the run information in the Gantt format:

1. In the **Gantt Chart** dialog box, click **Export to file.** The Save As dialog box opens.
2. Select the folder in which you want to save the file, and type a name for the file, and then click **Save.** A .gnt file is created in the folder you specified.

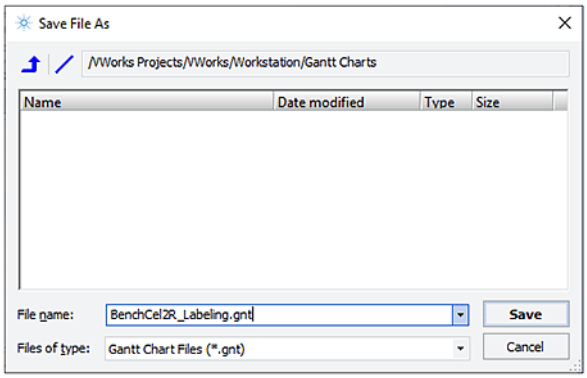
Importing and opening saved Gantt charts

To import the Gantt chart into Shared Services storage:

1. Click **File > Import Misc File**. The Select a Miscellaneous File for Import dialog box opens.
2. Select the **Gantt Chart Files (*.gnt)** file type, select the file, and click **Open**.

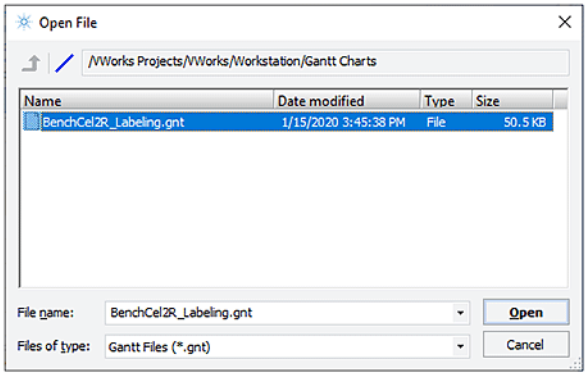


3 In the **Save File As** dialog box, select the storage location, verify the file name, and click **Save**.



To open an imported Gantt chart file:

1 Click **File > Open > Gantt**. The Open File dialog box opens.



2 Select the *.gnt file and click **Open**. The selected file opens in the Gantt Chart dialog box.

To import run information in Gantt format:

1 In the **Gantt Chart** dialog box, click **Import from file.** The Open dialog box opens.

2 Select the .gnt file, and then click **Open.** The run information appears in the Gantt Chart dialog box.

Related information

For information about...	See...
The System State Editor	"About the System State Editor" on page 730
Monitoring the overall run progress	"Monitoring the overall run progress" on page 329
Viewing logs	"Viewing logs" on page 714

Pausing the run

You can pause runs using one of the following methods:

- Pause all runs
- Pause selected runs in a runset

In addition, a protocol run will automatically pause when it reaches a User Message or Wait for User (Bravo) task. For details, see "Pausing at a User Message or Wait for User task" on page 338.

Pausing all runs

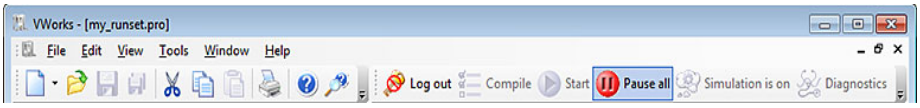
You can pause all runs that are in progress, and then continue the runs when you are ready. Pause all runs when you want to:

- Add or remove labware
- Add buffer to a reservoir
- Diagnose a problem
- Perform an operation that is not part of the protocol

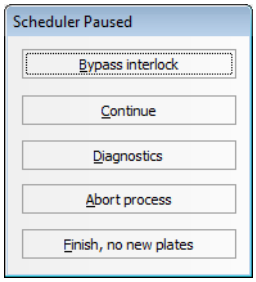
If you need to stop a run in an emergency, press the emergency-stop button on the pendant. See the device user guide for the procedure.

To pause all runs currently in progress:

1 In the **VWorks** window, click **Pause all** on the toolbar.



The Scheduler Paused dialog box opens.



2 If you moved or removed labware from the system, or if you fixed a device problem so that locations on the device are now available, select **Tools** > **System State Editor**. Change the process plate status and the device location status. For instructions, see "About the System State Editor" on page 730.

3 In the **Scheduler Paused** dialog box, click one of the following:

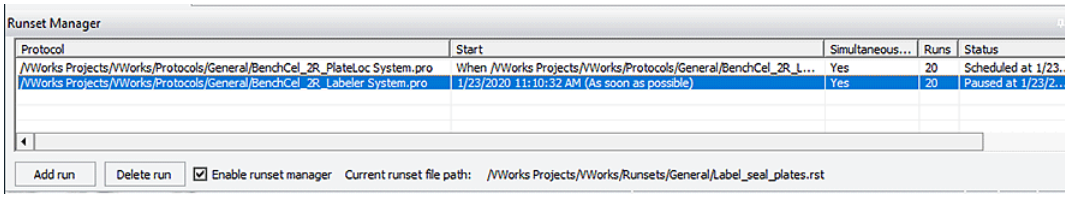
Command	Description
Continue	Resumes the protocol run. **IMPORTANT** When you pause one or more runs, the Orbital Shaking Station stops. Before you continue the run, open Teleshake Diagnostics to restart the shaking.
Diagnostics	Allows you to select the device diagnostic software to open and troubleshoot a problem or perform a manual operation. *Note:* If you click Diagnostics and open a device diagnostics dialog box, you will not be able to open the Bypass Interlock dialog box.
Abort process	Aborts the current command or task in the run. Select Abort if you have determined that the protocol run is not recoverable.
Finish, no new plates	Resumes the protocol run. Processes that are currently in progress will finish. However, no new labware will be delivered into the system.

Pausing selected runs in a runset

If you want to bypass the interlock and enter the system, use the Pause All command. See "Pausing all runs" on page 337. To stop a run in an emergency, use the emergency-stop button on the pendant. See the safety instructions for the system, workstation, or device for details.

To pause one protocol run in the Runset Manager:

1. In the **Runset Manager** tab, right-click the protocol you want to pause.
2. In the shortcut menu that appears, select **Pause this run.** The selected protocol is paused, and its status is shown in the Runset Manager.



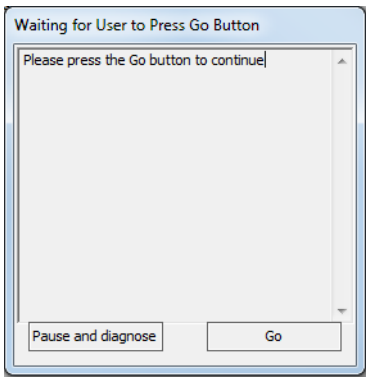
Alternatively, you can select **Temporarily prevent new plates from entering system** in the shortcut menu to allow other runs to resume. Processes that are currently in progress will finish. No new labware will be delivered into the system.

3. To resume the paused run, right-click the paused protocol in the **Runset Manager** tab, and then click **Resume this run**.

Pausing at a User Message or Wait for User task

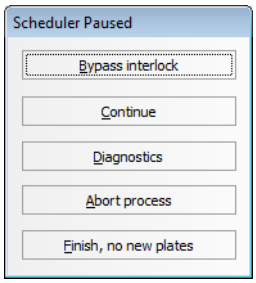
A run will automatically pause when it encounters a User Message or Wait for User task. These tasks pause the run to remind the operator to perform a task manually, such as refill a reservoir or replace labware.

When the system pauses, the message or Waiting for User dialog box appears.



To confirm that you want to pause or to diagnose a problem:

1 Click **Pause and Diagnose**. The Scheduler Paused dialog appears.



2 If you moved or removed labware from the system, or if you fixed a device problem so that locations on the device are now available, select **Tools** > **System State Editor**. Change the process plate status and the device location status. For instructions, see "About the System State Editor" on page 730.

3 In the **Scheduler Paused** dialog box, click one of the following:

Command	Description
Continue	Resumes the protocol run. **IMPORTANT** When you pause one or more runs, the Orbital Shaking Station stops. Before you continue the run, open Teleshake Diagnostics to restart the shaking.
Diagnostics	Allows you to select the device diagnostic software to open and troubleshoot a problem or perform a manual operation. *Note:* If you click Diagnostics and open a device diagnostics dialog box, you will not be able to open the Bypass Interlock dialog box.
Abort process	Aborts the current command or task in the run. Select Abort if you have determined that the protocol run is not recoverable.
Finish, no new plates	Resumes the protocol run. Processes that are currently in progress will finish. However, no new labware will be delivered into the system.

Related information

For information about...	See...
The System State Editor	"About the System State Editor" on page 730
Viewing logs	"Viewing logs" on page 714
Monitoring the protocol run	"Monitoring the overall run progress" on page 329
User Message task	"User Message" on page 682
Wait for User (Bravo) task	"Wait for User (Agilent Bravo)" on page 688

Stopping the run

If you want to stop a run and later continue the run, use one of the pause methods described in "Pausing the run" on page 337.

If you need to stop a run in an emergency, use the hardware Emergency Stop button. See the device user guide for the procedure.

CAUTION **You may not be able to resume a protocol run after you press the hardware Emergency Stop button. To recover the system after an emergency stop, see the device user guide.**

Related information

For information about...	See...
Pausing runs	"Pausing the run" on page 337
Starting runs	"Starting the protocol run" on page 312
Managing runsets	"Managing runsets" on page 317
Monitoring runs	"Monitoring the overall run progress" on page 329

Setting parameters for microplate-handling tasks

This chapter contains the following topics:

- "Centrifuge Process" on page 344
- "CentrifugeAuto" on page 353
- "Configure Static Labware" on page 361
- "Delid" on page 363
- "Dismount" on page 367
- "Incubate" on page 370
- "Mount" on page 374
- "Move to Location (Agilent Bravo)" on page 376
- "Place Plate" on page 378
- "Print and Apply" on page 381
- "Relid" on page 397
- "Reserve Location" on page 401
- "Rotate Stage (Agilent Microplate Labeler)" on page 403
- "Seal (Agilent PlateLoc)" on page 405
- "Waste" on page 408

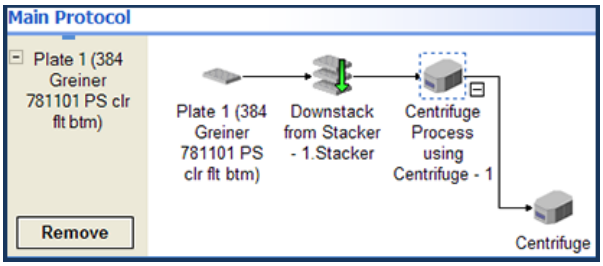
Centrifuge Process

Description

IMPORTANT

Agilent recommends that you use the CentrifugeAuto task instead of the Centrifuge Process task. The CentrifugeAuto task performs all the functions of the Centrifuge Process task but does not require a subprocess (see "CentrifugeAuto" on page 353). The Centrifuge Process task was used in protocols created in VWorks version 11.1 or earlier.

The Centrifuge Process task for the Centrifuge (Centrifuge Process (Centrifuge)) and the Centrifuge with Loader (Centrifuge Process (Centrifuge Loader)) indicates the start of a protocol subroutine that employs the Microplate Centrifuge (or Centrifuge). Within the subprocess, the software automatically adds the (Centrifuge) task to spin microplates according to the task parameter settings.



One microplate with a counterweight or two microplates can be spun at one time. Counterweights can be placed manually or robotically.

Task is available for...	Task is available in...
Centrifuge, or Centrifuge with Loader	Main Protocol

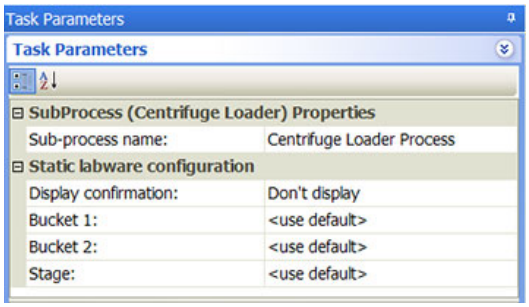
Task parameters

You set parameters for:

- *Centrifuge Process*. Contains the counterweight position information. For more information, see "Centrifuge Process" on page 344.
- *Centrifuge task*. Contains the spin parameters and counterweight selections. For more information, see "Centrifuge" on page 345.

Centrifuge Process

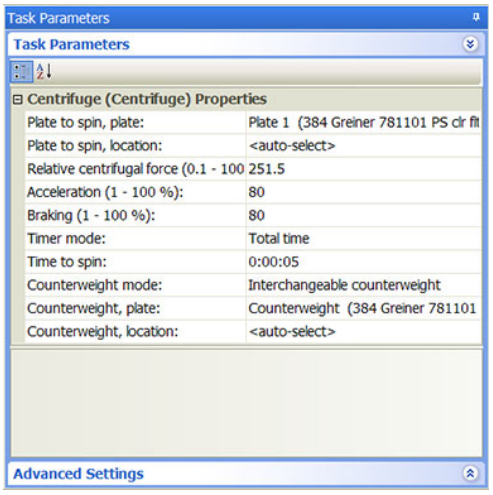
After you add the Centrifuge Process task (subprocess) at the desired point in the protocol, select the subprocess icon and set the following parameters in the **Task Parameters** area:



Parameter	Description
Sub-process name	The name of the subprocess. Select from the list of available subprocesses currently in the protocol.
Display confirmation	The option to display a message at the beginning of the protocol run to remind you to verify the physical locations of the labware match what you specified in the software.
Bucket 1/Bucket 2	The counterweight you want to use in the bucket location. You can select from the list of labware for either Bucket 1 or Bucket 2. The counterweight selection depends on the counterweight mode you select in the Centrifuge task: • If you are using the **Interchangeable counterweight** or **Use 2 protocol plates** counterweight mode, select **<use default>.** • If you are using the **Fixed Counterweight** mode, select a labware for either Bucket 1 or Bucket 2. For a description of the counterweight modes, see "Centrifuge" on page 345.
Stage (*Centrifuge with Loader only*)	The stage at the Centrifuge Loader.

Centrifuge

When you add the Centrifuge Process task (subprocess) at the desired point in the protocol, the Centrifuge task is automatically added. Select the Centrifuge task and set the following parameters in the **Task Parameters** area:



Parameter	Description
Plate to spin, plate	The microplate that will be spun. *Note:* Select the counterweight microplate only if you plan to use the **Fixed counterweight** mode.
Plate to spin, location	The device location to use. For example, you can select a centrifuge bucket. <auto-select> automatically places the labware at the first-available or appropriate location for the task. Make sure the **Plate to spin, location** selection does not conflict with the **Counterweight, location** selection.
Relative centrifugal force (0.1–1006.2 × g)	The rotor velocity, as a multiple of gravity.
Acceleration (1–100%)	The rate of centrifugation, as a percent of maximum acceleration.
Braking (1–100%)	The deceleration of the centrifuge, as a percent of maximum deceleration.
Timer mode	How the specified spin time is implemented: • **Total time.** The specified spin time includes acceleration and braking. • **Time at speed.** The specified spin time does not include acceleration and braking.
Time to spin	The length of time to spin the microplates in the desired mode.

Parameter	Description
Counterweight mode	The counterweight method used. Select one of the following: • **Fixed counterweight.** The counterweight is manually placed in the centrifuge before starting the run. The counterweight remains in the centrifuge during the entire run. You must remember to place the counterweight in the device before the run and remove it after the run is finished. **IMPORTANT** Be sure to select the counterweight in the Centrifuge Process Task Parameters area. See "Task parameters" on page 344. • **Interchangeable counterweight.** The counterweight is on a platepad or in the Labware MiniHub. The robot will move it into the centrifuge during the run and return it to the platepad when the run is finished. If you select this mode, use the **Counterweight, plate** parameter to select the counterweight you want to use. • **Use 2 protocol plates.** Instead of a counterweight, a second process plate is used during the run. You must process two or more microplates in this mode. During the run, the robot places the first two microplates into the centrifuge. After spinning, the robot replaces the two microplates with the next pair, and so on. If you are processing an odd number of microplates, and three microplates are remaining, the robot will not remove the second from the last microplate so that the last microplate is paired. If you are processing an odd number of microplates, and: – A process contains only one Centrifuge Process task and it uses this counterweight mode, set the **Simultaneous plate** parameter to 2. – A process contains two or more Centrifuge Process tasks, and they all use this counterweight mode, set the **Simultaneous plates** parameter to 3 or greater.

Parameter	Description
Counterweight, plate	The microplate or labware you want to use as the counterweight. This parameter is only available if you selected the **Interchangeable counterweight** method. If the list of counterweights do not appear, make sure: • You have added a device such as the Platepad for the counterweight. See "Adding devices" on page 26 for instructions. • You have configured the labware correctly. See "Configuring labware" on page 53 for instructions.
Counterweight, location	The device location to use. For example, you can select a centrifuge bucket. <auto-select> automatically places the labware at the first-available or appropriate location for the task. Make sure the **Counterweight, location** selection does not conflict with the **Plate to spin, location** selection.

Device selection

You must select a device for Centrifuge Process tasks. If you have multiple devices of the same type, you can:

- Prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.
- Set up a backup pool. If the primary device encounters an error, the software will automatically use the next device in the list. However, if all of the devices in the list are in an error state, the software will automatically use the device in the backup pool.

IMPORTANT

The multiple devices must have the same setup and configuration.

After adding the Centrifuge Process task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

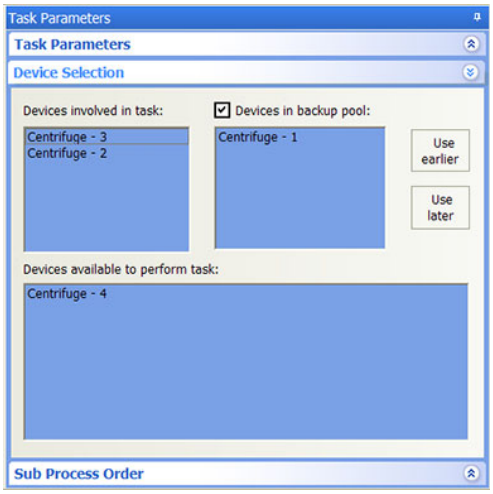
1. Double-click the desired device in the **Devices available to perform task** area to move it to the **Devices involved in task** area. If you have multiple devices of the same type, you can move them to the **Devices involved in task** area.
2. If you have multiple devices in the **Devices involved in task** area, select a device, and then click **Use earlier** or **Use later** to prioritize it.

3 *Optional.* Select backup devices to use in case the primary device in the **Devices involved in task** area encounters an error.

a Select **Devices in backup pool.**

b Drag one or more devices from the **Devices available to perform task** area to the **Devices in backup pool** area.

c If you have multiple devices in the backup pool, select a device in the **Devices in backup pool** area, and then click **Use earlier** or **Use later** to prioritize it.



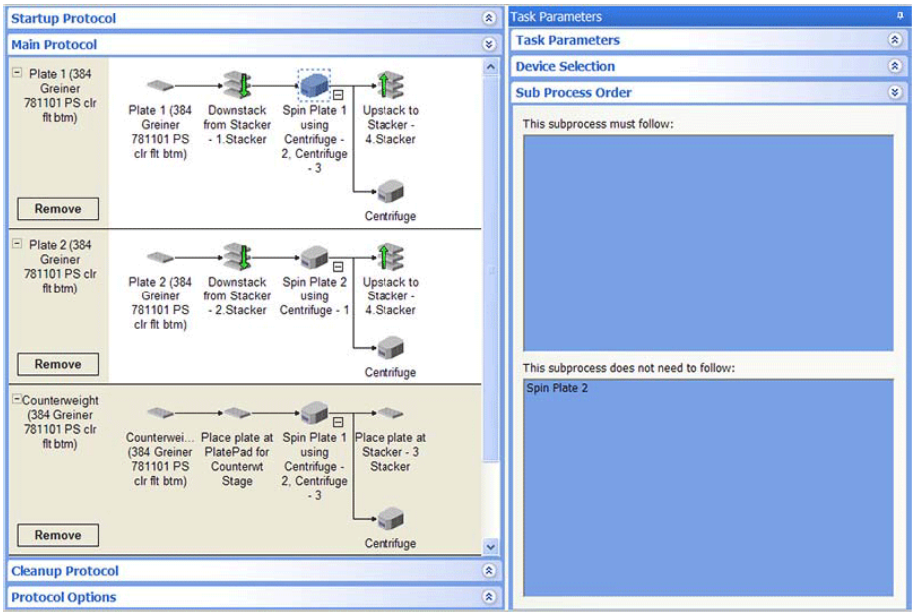
Subprocess order

If more than one subprocess uses the same configured labware, and the subprocesses are in different protocol processes, you can specify the sequence in which the subprocesses will be performed.

To specify the sequence in which the subprocesses will be performed on the same configured labware:

1 In the protocol process, select the sub-process that contains the task that uses the configured labware.

In the following example, the Spin Plate 1 process is selected.



2 In the **Task Parameters** area, click **Sub Process Order.**

3 In the **Sub Process Order** area, double-click the subprocess names to rearrange the order.

In the example shown in step 1, the order is Spin Plate 1 and Spin Plate 2.

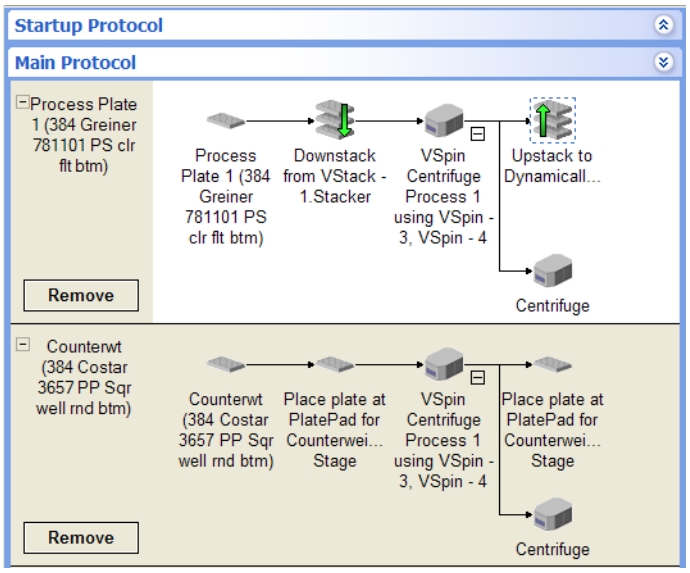
Example: Interchangeable counterweight mode

Goal

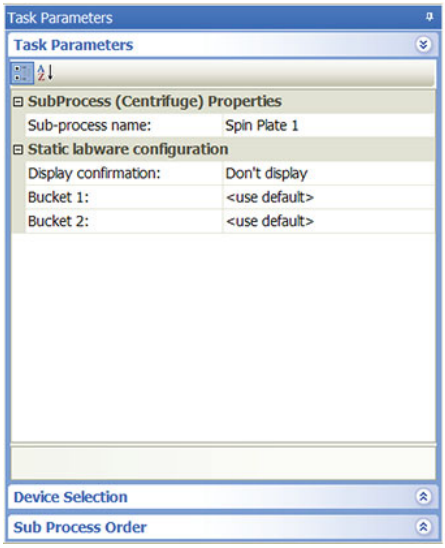
Downstack microplates from a BenchCel stacker, spin the microplates in the Microplate Centrifuge, and then upstack the microplates to an available BenchCel stacker. Use the counterweight that is stored on a designated platepad.

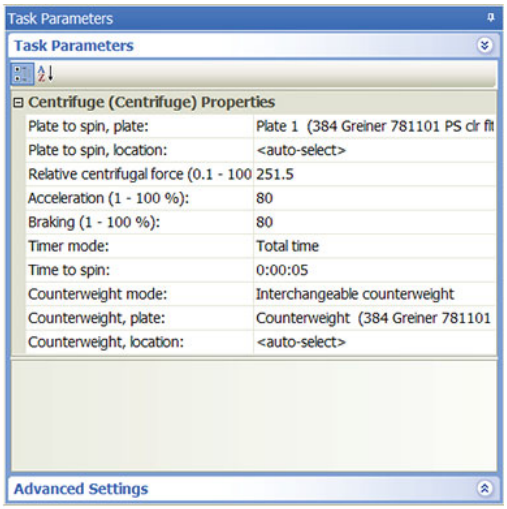
Implementation

(1) Add a process for microplates that will be downstacked, spun, and upstacked. Add a Downstack, Centrifuge Process, and Upstack task in the order shown. (2) Configure the counterweight on a platepad and call it Counterweight.



The task parameters for the Centrifuge Process and the Centrifuge task are shown. Notice that the Bucket n parameters are set to <use default>, because the Interchangeable counterweight mode is selected.





Because the robot will move the counterweight from the designated platepad to the centrifuge during the run, select the Interchangeable counterweight mode. For the **Counterweight, plate** parameter, select Counterweight.

Note: The robot will move the counterweight back to the platepad when the run is finished.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide

For information about...	See...
Adding tasks in a protocol	“Adding and deleting tasks” on page 67
Configuring labware	“Configuring labware” on page 53
Microplate-storage tasks	“Setting parameters for microplate storage tasks” on page 411
Liquid-handling tasks	“Setting parameters for liquid-handling tasks” on page 449
Scheduling tasks	“Setting parameters for scheduling tasks” on page 639

CentrifugeAuto

Description

Note: If you want to manage counterweights and counterweight modes using JavaScript, see the examples in "Using JavaScript with the CentrifugeAuto task" on page 136.

Note: The CentrifugeAuto task (CentrifugeAuto) performs all the functions of the Centrifuge Process task but does not require a subprocess.

The CentrifugeAuto task (CentrifugeAuto) can be used with the Microplate Centrifuge (Agilent Centrifuge) and Centrifuge with Loader (Agilent Centrifuge Loader) devices to spin microplates according to the task parameter settings. The CentrifugeAuto task enables you to spin one microplate with a counterweight or spin two microplates from a protocol process, at the same time. Counterweights can be placed manually or robotically. When using counterweights, you can use multiple counterweights for protocols with multiple processes. This is accomplished by using the Configure Labware feature to manage the centrifuge tasks.

Task is available for...	Task is available in...
Centrifuge	Main Protocol
Centrifuge with Loader	Main Protocol

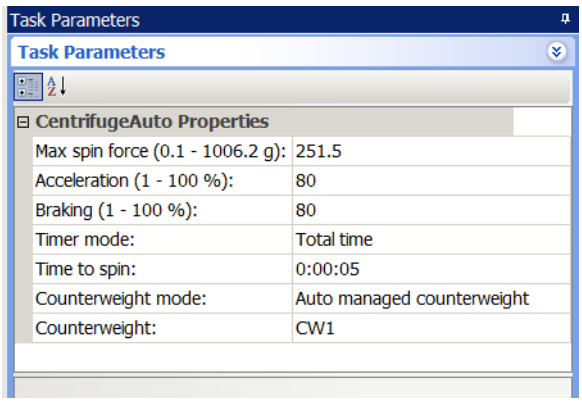
The CentrifugeAuto task cannot be used in the following cases:

- Within a Loop task
- Within a Spawn process as a subroutine, as this will result in a deadlock
- Within a Macro
- As the first task in a process

You cannot assign multiple devices to a single CentrifugeAuto task.

Task parameters

After adding the CentrifugeAuto task in the main protocol, set the **CentrifugeAuto Properties** in the **Task Parameters** area:



Parameter	Description
Max spin force (0.1 - 1006.2 × g)	The rotor velocity, as a multiple of gravity.
Acceleration (1–100%)	The rate of centrifugation, as a percent of maximum acceleration.
Braking (1–100%)	The deceleration of the centrifuge, as a percent of maximum deceleration.
Timer mode	The way the spin time is implemented: • **Total time.** The specified spin time includes acceleration and braking. • **Time at speed.** The specified spin time does not include acceleration and braking.
Time to spin	The length of time to spin the microplates in the desired mode.
Counterweight mode	The method used to keep the Centrifuge balanced during a spin.
Fixed counterweight	The manual mode for placing the counterweight. Place the counterweight in the device before the run (in bucket 2) and remove it after the run is finished. **CAUTION Do not use the Fixed counterweight mode with other counterweight modes on the same device. Running a protocol with mixed counterweight modes will result in a compiling error. Ignoring the error and continuing to run the protocol could result in a collision.**
Auto managed counterweight	A counterweight managed as a **Configure Labware** process. The robot will move the counterweight from a platepad or Labware MiniHub (specified by the plate process) into the Centrifuge during the run and return it to the platepad when the run is finished. If you select this mode, use the **Counterweight** parameter to select the **Configure Labware** process that corresponds to the counterweight you want to use. **Guidelines for use** To avoid deadlocks when there are two or more CentrifugeAuto tasks in a protocol: If the tasks are placed before pipetting tasks for both source and destination processes, and if the destination plate is spun first, set up an additional temporary-storage location for the destination plate. This mode can be used in combination with the Use 2 protocol plates mode in the same protocol. See "Guidelines for use with two modes (Auto managed and Use 2 protocol plates)" on page 355 for additional guidelines.

Parameter	Description
Use 2 protocol plates	A second process plate used as the counterweight. During the run, the robot places the first two microplates into the centrifuge. After spinning, the robot replaces the two microplates with the next pair, and so on. If you are processing an odd number of microplates, and three microplates are remaining, the robot will not remove the second from the last microplate so that the last microplate is paired. **Guidelines for use** To avoid deadlocks, whenever using this task, set **Simultaneous plates** to 2 or greater and run the protocol at least twice. To avoid deadlocks when there are two or more CentrifugeAuto tasks in a protocol: • If the tasks are within the same process and are using different devices and if Simultaneous plates is set to 2, run an even number of microplates. • If the tasks are within the same process and are using the same device and if you are running an odd number of plates, set up an additional temporary storage location to accommodate the odd plate. • If the tasks are placed before pipetting tasks for both source and destination processes, and if the destination plate is spun first, set up an additional temporary-storage location for the destination plate. • If using the same Centrifuge with Loader, ensure the tasks are not adjacent to each other. *Note:* This is true only for the Centrifuge with Loader and not for the Centrifuge. This mode can be used in combination with the Auto managed mode in the same protocol. See "Guidelines for use with two modes (Auto managed and Use 2 protocol plates)" on page 355 for additional guidelines.
Counterweight	A configured labware used as the counterweight. This parameter is available only if you selected the **Auto managed counterweight** mode. Tip boxes and reservoirs, or any other labware not compatible with the Microplate Centrifuge or Centrifuge with Loader, cannot be used as counterweights. If the list of counterweights do not appear, make sure you have configured the labware correctly. See "Example: Auto managed counterweight mode" on page 356 for instructions.

Guidelines for use with two modes (Auto managed and Use 2 protocol plates)

Use the following guidelines to avoid deadlocks when the protocol has two or more CentrifugeAuto tasks, and one task uses the Auto managed mode and another uses the Use 2 protocol plates mode.

Multiple CentrifugeAuto tasks using the same device

If the tasks are using the same device, run an even number of plates.

A Reorder task is between two CentrifugeAuto tasks

- If using the Reorder task between two CentrifugeAuto tasks with the same device, and if placing the auto-managed CentrifugeAuto task before the Reorder task, set the **Number of plates** to store in the Reorder task to an even number.
- If using the Reorder task between two CentrifugeAuto tasks and if placing the Use 2 protocol plates mode CentrifugeAuto task before the Reorder task, set the **Number of plates** to store in the Reorder task to an even number.

A long Incubation task is between two CentrifugeAuto tasks

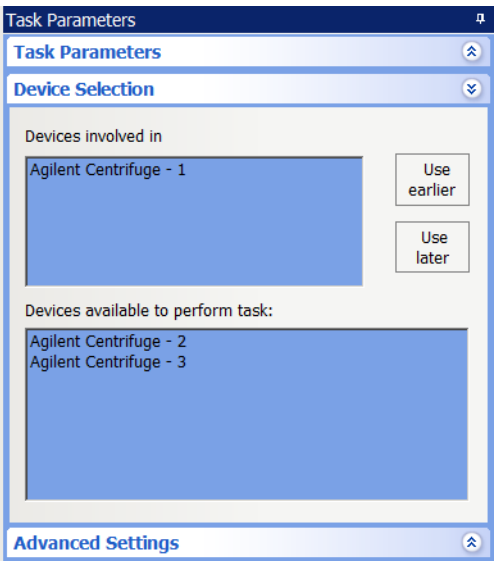
If placing a long Incubation task between two CentrifugeAuto tasks that use the same device, and if the Auto managed CentrifugeAuto task is before the Incubation task, set the number of incubation locations to an even number.

Device selection

You must select a device for each CentrifugeAuto task. After adding the CentrifugeAuto task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

Double-click the desired device in the **Devices available to perform task** area to move it to the **Devices involved in task** area. Only one device can be assigned to a given task.



Example: Auto managed counterweight mode

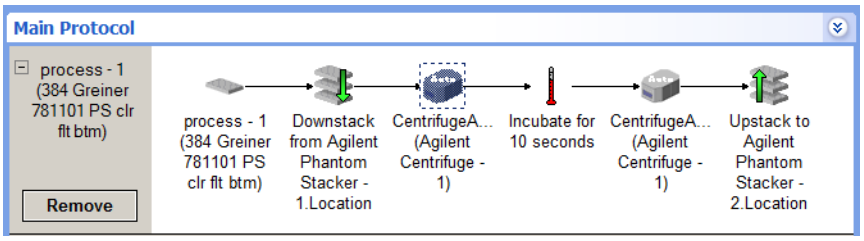
Goal

Perform two spins on one microplate with a different task between spins, in this case, an incubate task. Both spins are performed with the same device and each spin uses a different counterweight.

Implementation

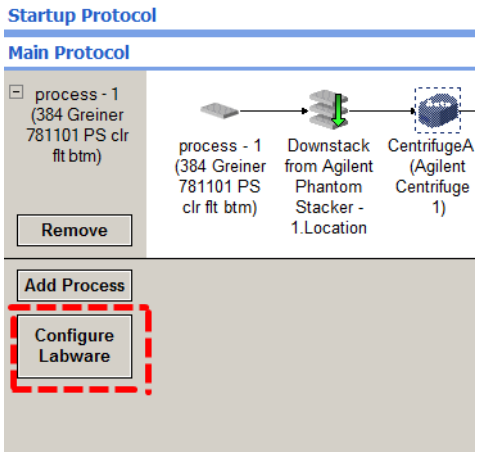
1 Create a protocol process that includes a Downstack, Spin, Incubate, a second Spin, and an Upstack task.

2 Select the parameters in the **Task Parameters** area for the Downstack, Incubate, and Upstack tasks.



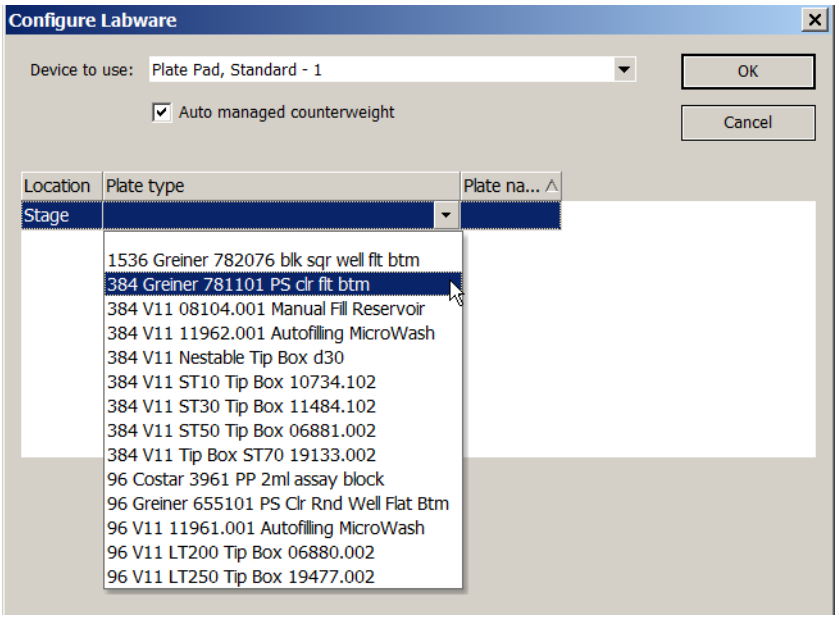
3 Create a Configure Labware process for each counterweight as follows:

a Click **Configure Labware**.

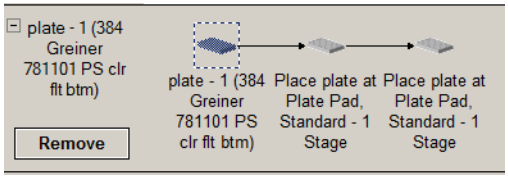


b In the **Configure Labware** dialog box, do the following:

- Select the location of a counterweight from the **Device to use** list.
- Select **Auto managed counterweight**.
- Select the microplate from the **Plate type** list.
- Click **OK**.

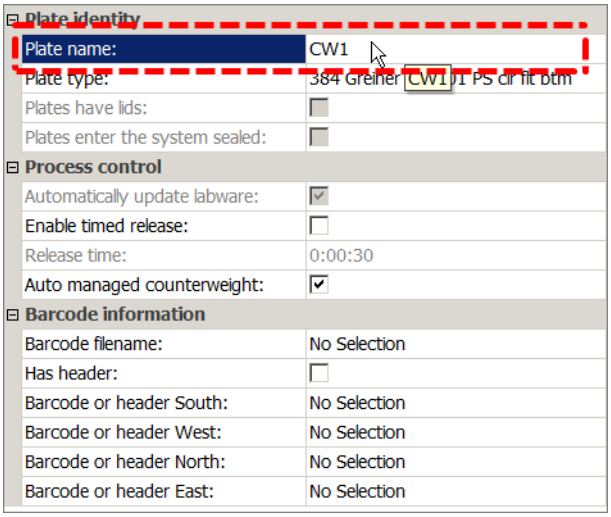


The configure labware process appears.



c In the **Task Parameters** area, type a name for the Configure Labware process in the **Plate name** field (in this case, **CW1**).

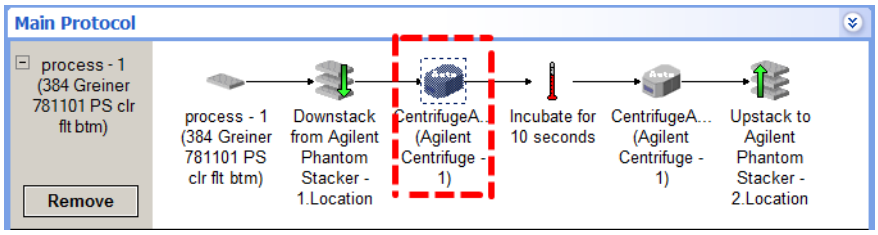
Note: When the **Auto managed counterweight** option is selected, the **Simultaneous plates** and **Automatically update labware** options are not available.



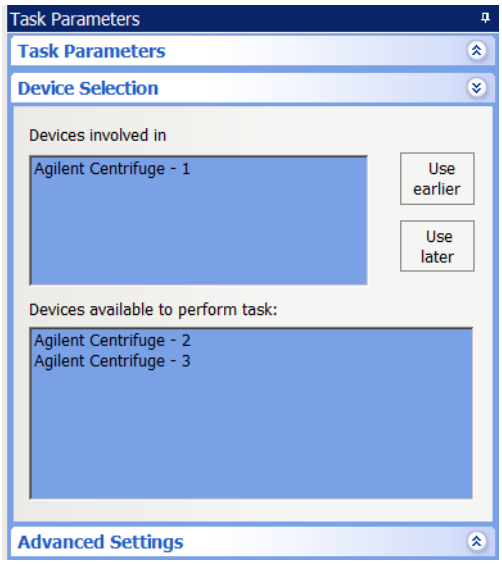
d Repeat steps 3a through 3c for the second counterweight.

4 Select the device for the **CentrifugeAuto** tasks:

a Select the first **CentrifugeAuto** task in the process.



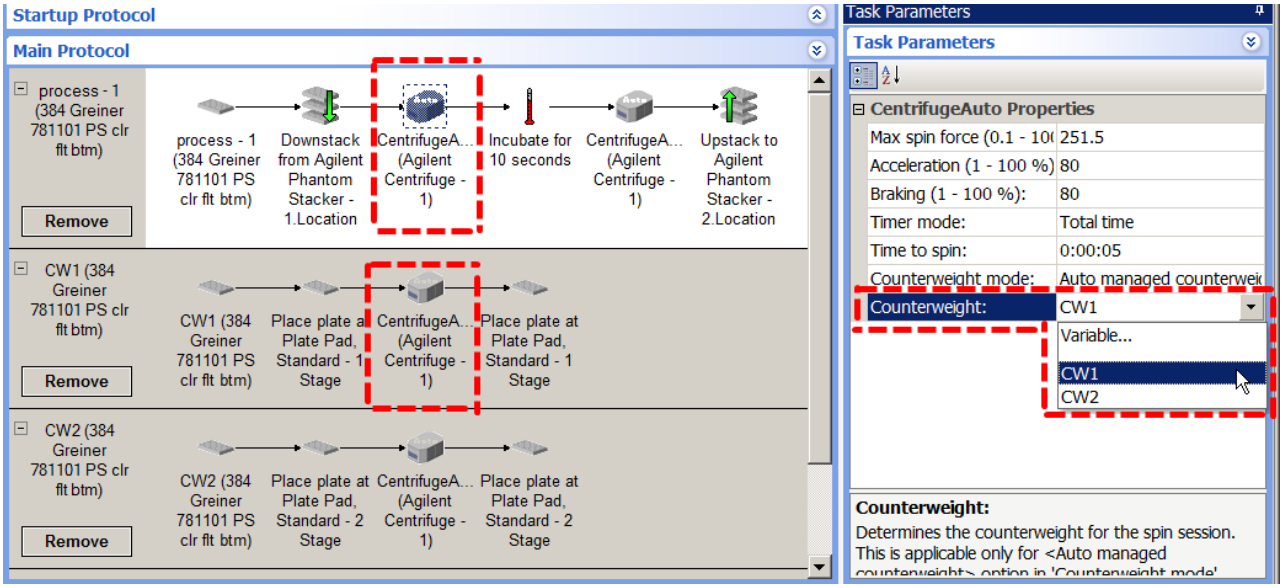
b In the **Device Selection** area, drag **Agilent Centrifuge-1** to the **Devices involved in task** area.



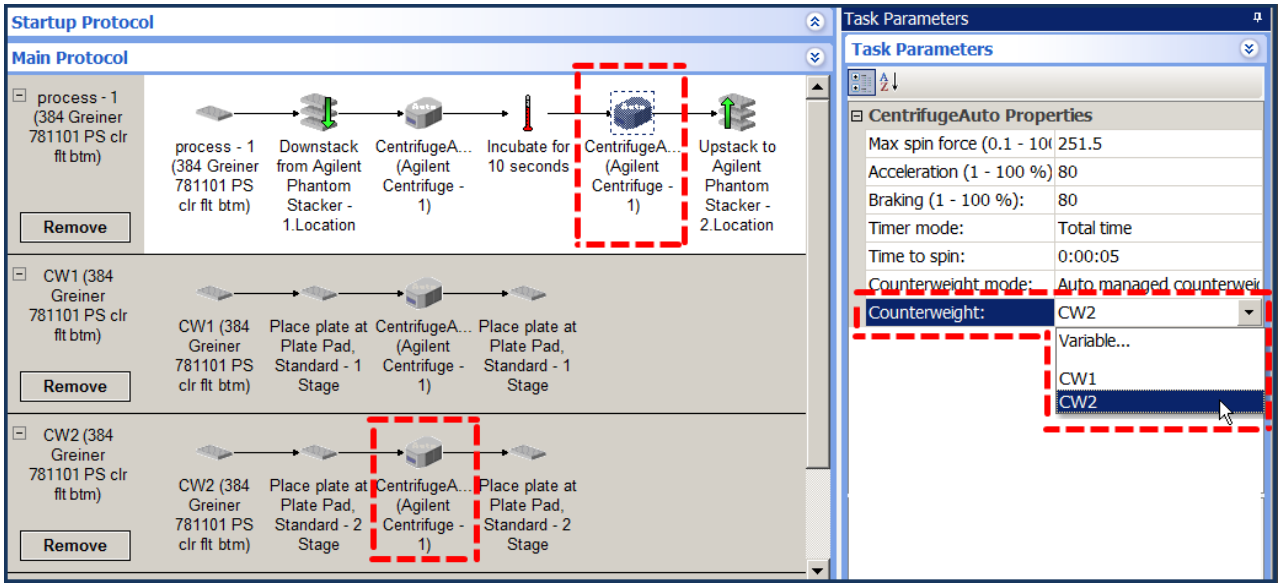
c Repeat steps 4a and 4b for the second **CentrifugeAuto** task.

5 When setting the parameters of the first **CentrifugeAuto** task, select the **Auto managed counterweight** mode, and then select **CW1** from the **Counterweight** list.

Note: A CentrifugeAuto task is automatically added to the Configure Labware process chosen for the counterweight.



6 When setting the parameters of the second **CentrifugeAuto** task, select the **Auto managed counterweight** mode, and then select **CW2** from the Counterweight list.



7 Before you start the run, make sure the counterweights are at their starting locations.

Plate movement

When the protocol runs, the robot will place the downstacked microplate in bucket 1 of the centrifuge and the counterweight, CW1, into bucket 2. Counterweight CW1 will be swapped out for counterweight CW2 after the microplate is incubated and after it is placed back into bucket 1 of the Centrifuge. Both counterweight plates will automatically be placed back on their starting locations at the end of the process.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Configuring labware	"Configuring labware" on page 53
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Configure Static Labware

Description

IMPORTANT

Although the VWorks software supports the static labware configuration in protocols, Agilent recommends that you use the concept and procedure in "Configuring labware" on page 53 when writing new protocols.

Available only in the Startup protocol, the Configure Static Labware task () allows you to assign static labware to locations on a device. The labware will remain at these locations during the protocol.

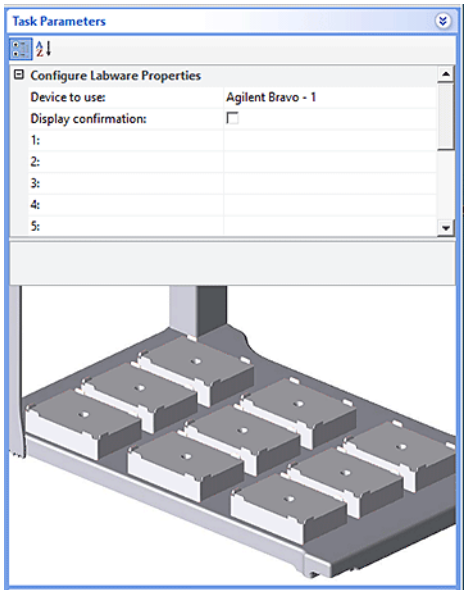
Note: You can also configure static labware in the device subprocess task in the main protocol. Configure labware in the device subprocess if you want to override the labware configuration in the startup protocol. Configure labware in a Startup Protocol if the labware configuration will be used in all the Main Protocol subprocesses.

In general, you configure static labware before the first task in a protocol. If you have multiple processes in the protocol, configure the labware once before the first task of the first process.

Task is available for...	Task is available in...
Any device	Startup Protocol

Task parameters

After adding the Configure Labware task in the startup protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Device to use	The device on which the labware is placed. Select from the list of available devices.
Display confirmation	The option to display a message to check the physical placement of the labware against the assignment in the software.
Device location	The list of locations on the device. For example, if you select Agilent Bravo as the device to use, deck locations 1 through 9 appear in the list. Click the device location, and then select the labware type.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Subprocess task	"SubProcess (Agilent Bravo)" on page 450
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

For information about...	See...
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Delid

Description

The Delid task () removes lids from microplates. You can use the Delid task to remove and discard a lid, or you can pair it with the Relid task to retain the removed lid and later place the lid back on the microplate.

Note: You can only delid process plates and configured labware. You cannot delid static labware.

Task is available for...	Task is available in...
BenchCel Microplate Handler	Startup Protocol Main Protocol Cleanup Protocol
Bravo Platform	Startup Protocol Main Protocol, Bravo Subprocess Cleanup Protocol

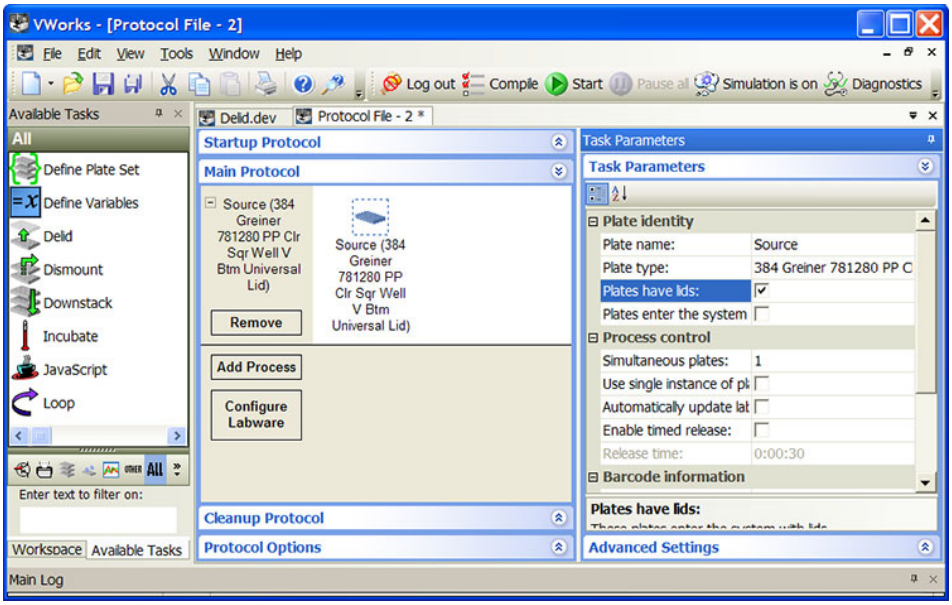
Requirements

Labware entry

In the Labware Editor, make sure the labware you want to delid has lid specifications in the **Plate Properties** tab. For detailed instructions, see the *VWorks Automation Control Setup Guide*.

Plate parameters

When you set parameters for the process plate, make sure you select the **Plates have lids** option. For detailed instructions, see "Setting plate parameters" on page 59.



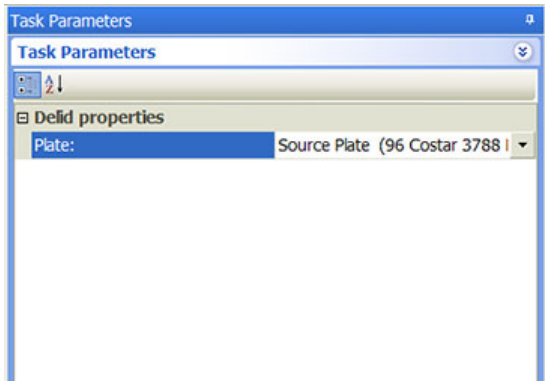
Bravo Platform

To use the Delid task, you must first configure the Lid Hotel in Bravo Diagnostics. For detailed instructions, see the *Bravo Platform User Guide*.

Task Parameters

Bravo Platform

To delid microplates on the Bravo Platform, you add the Delid task within the Bravo Subprocess. After adding the Delid task at the desired point in the subprocess, select the microplate you want to delid in the **Task Parameters** area, as the following example shows.



Parameter	Description
Plate	The microplate that will be delidded.

Device selection

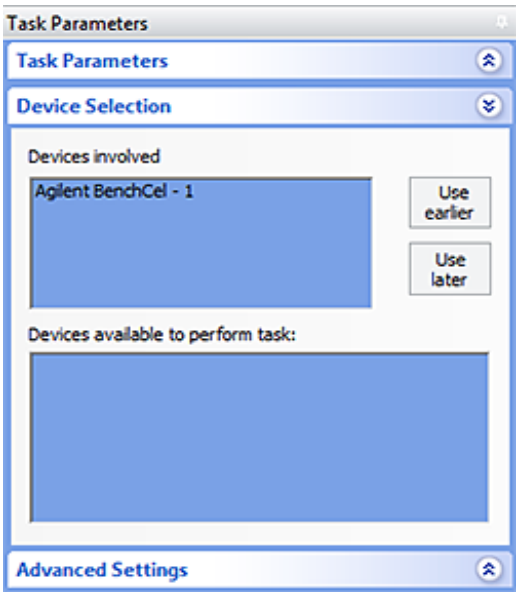
You do not need to select a device when the Delid task is within a Bravo Subprocess, because the software assumes you want to use the Bravo Platform.

Note: If you have multiple devices of the same type, you can prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available. The multiple devices must have identical setup and configuration.

After adding the Delid task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1. Double-click the desired device in the **Devices available to perform task** area to move it to the **Devices involved in task** area. If you have multiple devices of the same type, you can move them to the **Devices involved in task** area.
2. If you have multiple devices in the **Devices involved in task** area, select a device, and then click **Use earlier** or **Use later** to prioritize it.



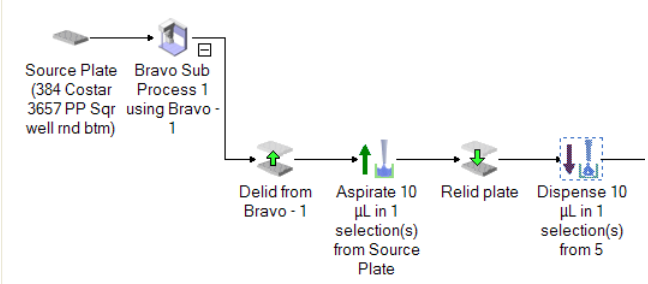
Example: Delid and relid microplates

Goal

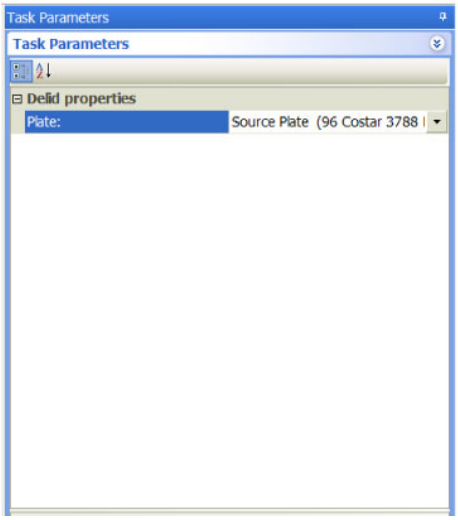
On the Bravo Platform, remove a lid from the Source Plate for an Aspirate task. After the task is finished, relid the microplate.

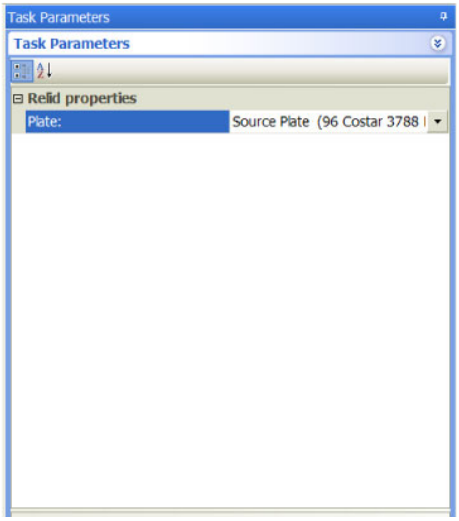
Implementation

Within the Bravo Subprocess, add a Delid task before the Aspirate task to remove the lid. Add a Relid task after the Aspirate task to place the lid back on the microplate.

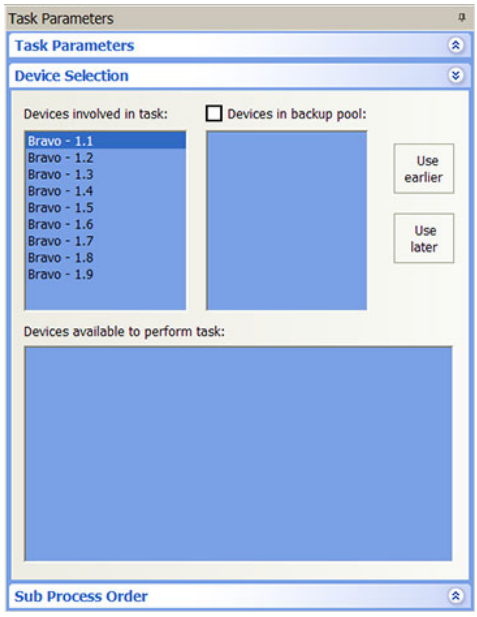


The task parameters for the Delid and Relid tasks are:





The device selection for the Delid task is as follows. Note that you do not need to select a device for the Relid task.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Relid task	"Relid" on page 397
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Dismount

IMPORTANT

The Dismount task is always paired with the Mount task. Before adding the Dismount task, review the description for the Mount task. See "Mount" on page 374.

The Dismount task () removes labware, such as a filter microplate, from the top of an elution microplate or reservoir that is resting on a platepad. Always paired with a Mount task, the Dismount task uses information that is specified in the Mount task.

For example, you can use the Mount task to place a filter microplate on a reservoir that catches the elution. After the centrifuge process, you can use the Dismount task to remove the filter microplate from the reservoir and discard it.

Task is available for...	Task is available in...
Platepad	Main Protocol

Task parameters

The Dismount task does not have task parameters.

Example: Dismount a filter microplate from a collection microplate

Goal

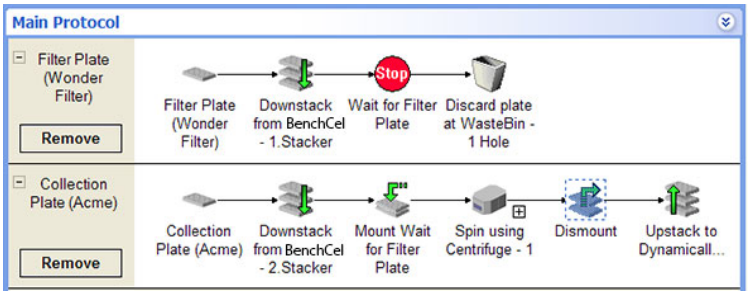
Downstack a collection microplate from a BenchCel stacker and place it on a platepad. Downstack a filter microplate from another BenchCel stacker and place it on top of the collection microplate. Centrifuge the assembly (filter microplate with collection microplate). Remove the filter microplate from the collection microplate and discard it. Upstack the collection microplate containing the elution for future processing.

Implementation

Add one process for the collection microplate and another process for the filter microplate. See "Example: Mount a filter microplate on a reservoir that collects elution" on page 375 for details about the Mount and Wait For tasks.

In the filter microplate process, add a Waste task after the Wait For task. The Waste task discards the filter microplate after the centrifuge is finished.

In the collection microplate process, add the following tasks after the Mount task: Centrifuge, Dismount (to remove the filter microplate from the collection microplate), and Upstack (to store the collection microplate containing the elution).



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Mount task	"Mount" on page 374
Downstack task	"Downstack" on page 414
Upstack task	"Upstack" on page 443
Waste task	"Waste" on page 408
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Incubate

Description

The Incubate task (Incubate) moves labware to a location, leaves it there for a specified time period, and then moves it from the location. The number of labware that can be incubated simultaneously is limited by the number of locations that are available for holding labware.

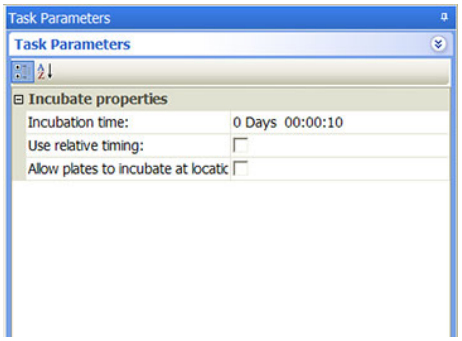
Note:

- To incubate a labware in a storage device such as a Labware MiniHub, use the Storage Incubate task.
- To incubate labware on the Bravo Platform, use the Reserve Location task.

Task is available for...	Task is available in...
Platepad	Main Protocol

Task parameters

After adding the Incubate task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Incubation time	The length of time to incubate the labware. *Note:* The actual incubation period might be longer than the incubation time you specify. The actual incubation period is affected by the scheduling and operating speed of the robot.
Use relative timing	The option to start timing the incubation after the previous labware has finished incubating. *Note:* The option can result in a longer incubation period than specified. Select this option to prevent bottlenecking at a downstream device. Alternatively, use the **Release time** parameter when setting process plate properties.

Parameter	Description
Allow plates to incubate at locations with BCRs	The option to permit a labware to be placed at a location with a barcode reader. The barcode reader is not used during the Incubate task.

Device selection

You must select a device for Incubate tasks. If you have multiple devices of the same type, you can:

- Prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.
- Set up a backup pool. If the primary device encounters an error, the software will automatically use the next device in the list. However, if all of the devices in the list are in an error state, the software will automatically use the device in the backup pool.

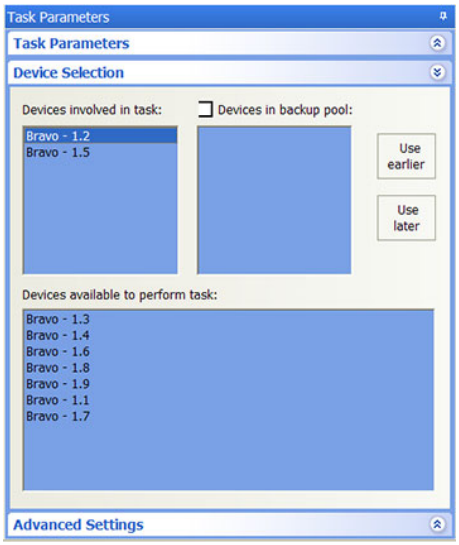
IMPORTANT

The multiple devices must have the same setup and configuration.

After adding the Incubate task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1. Double-click the desired device in the **Devices available to perform task** area to move it to the **Devices involved in task** area. If you have multiple devices of the same type, you can move them to the **Devices involved in task** area.
2. If you have multiple devices in the **Devices involved in task** area, select a device, and then click **Use earlier** or **Use later** to prioritize it.
3. *Optional.* Select backup devices to use in case all of the devices in the **Devices involved in task** area encounter an error.
   a. Select **Devices in backup pool.**
   b. Drag one or more devices from the **Devices available to perform task** area to the **Devices in backup pool** area.
   c. If you have multiple devices in the backup pool, select a device in the **Devices in backup pool** area, and then click **Use earlier** or **Use later** to prioritize it.



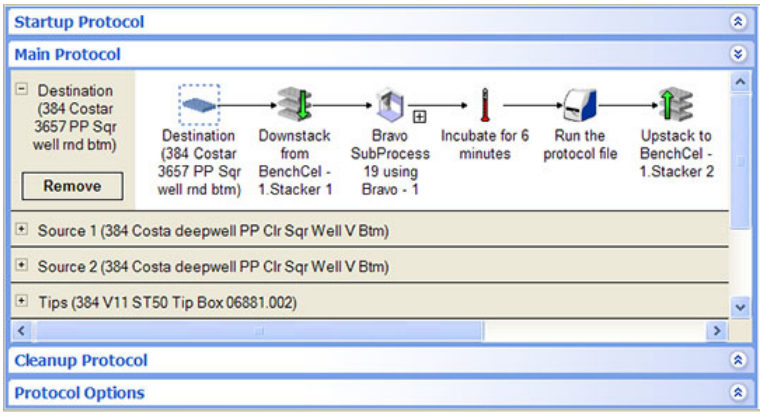
Example: Incubate a microplate after mixing two reagents

Goal

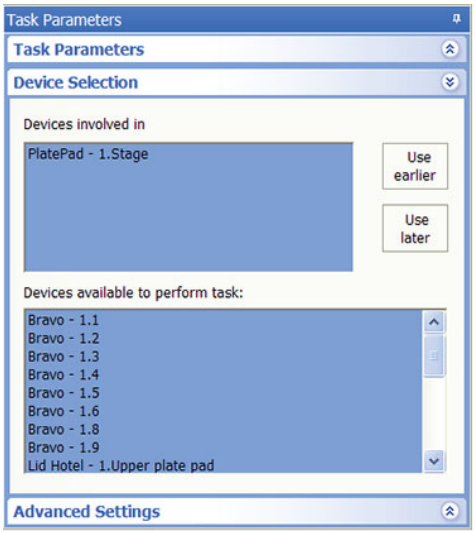
Aspirate contents from two source microplates, dispense into a destination microplate. Incubate the mixture for at least 6 minutes, and then read the microplate.

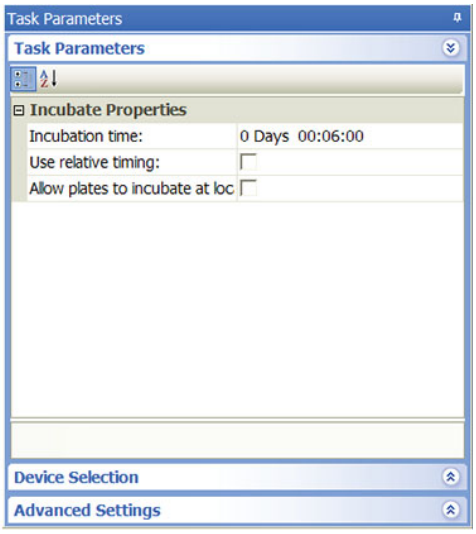
Implementation

In the protocol, add an Incubate task after the liquid-handling tasks as shown in the following example.



For the Incubate task, select platepad-1 for the incubation location. Set Incubation time to 6 minutes in the Task Parameters area as shown.





Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Process plate properties	"Setting plate parameters" on page 59
Storage Incubate task	"Storage Incubate" on page 434
Reserve Location task	"Reserve Location" on page 401
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Mount

Description

The Mount task (Mount) places a labware, such as a filter microplate, on top of a collection microplate or reservoir that is resting on a platepad. The Mount task works with the Wait For task and is paired with the Dismount task.

For example, you can use the Mount task to place a filter microplate on a reservoir that catches the elution. After the centrifuge process, you can use the Dismount task to remove the filter microplate from the reservoir and discard it.

The Mount task becomes available when you add a Platepad device in the device file. The task can be used only in the Main Protocol.

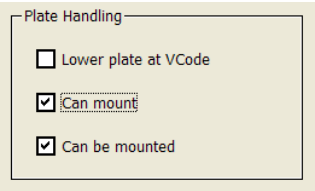
Task is available for...	Task is available in...
Platepad	Main Protocol

Requirements

Labware definition

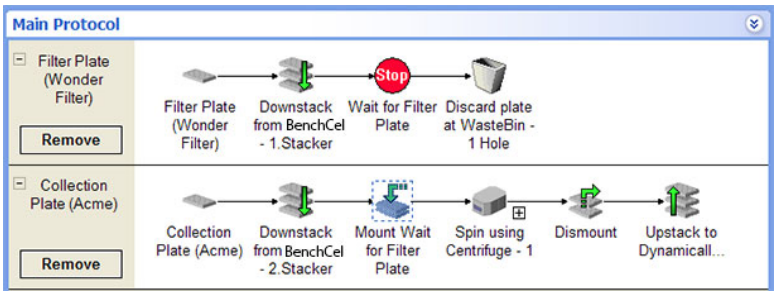
The labware involved in the Mount task must have the following Plate Handling properties in the Labware Editor:

- *Can be mounted.* Select the **Can be mounted** property to define microplates or reservoirs that will be mounted (will be on the bottom).
- *Can mount.* Select the **Can mount** property to define microplates that will mount on top of the elution microplate or reservoir.



Process setup

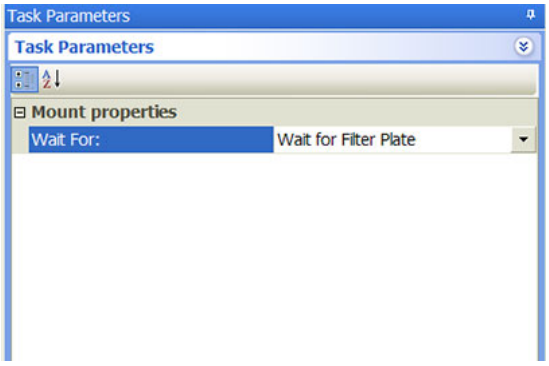
Each labware involved in the Mount task should be set up as independent processes in the protocol. For example, if you want to place a filter microplate on top of a reservoir or collection microplate that catches the elution, you should create two processes as shown in the following example:



Notice that a Wait For task is added to the Filter Plate process at the point where the mount task would occur. A Mount task is added to the Collection Plate process at the desired point. See "Example: Mount a filter microplate on a reservoir that collects elution" on page 375 for a detailed description.

Task parameters

After adding the Mount task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Wait for	The name of the Wait For task that you want to pair with this Mount task. Select from the list of Wait For tasks. The list is based on the Wait For tasks that have been added in the protocol.

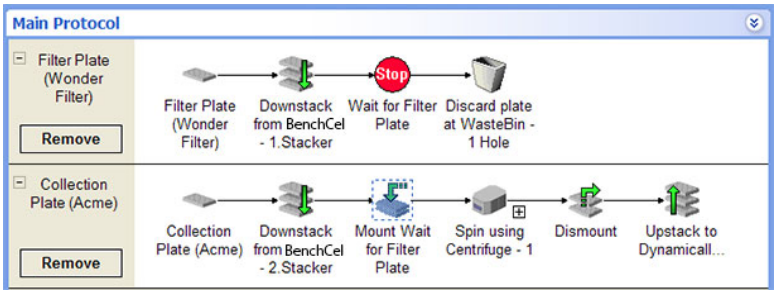
Example: Mount a filter microplate on a reservoir that collects elution

Goal

Downstack a reservoir (Collection Plate) from a BenchCel stacker and place it on a platepad. Downstack a filter microplate (Filter Plate) from another BenchCel stacker and place it on top of Collection Plate.

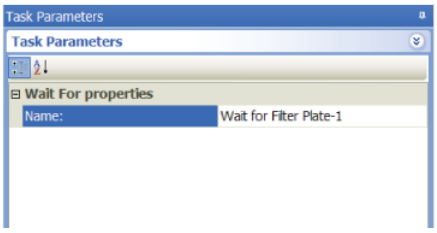
Implementation

Create one process for the Collection Plate and another process for the Filter Plate. In the Filter Plate process, add a Wait For task after the Downstack task. In the Collection Plate process, add a Mount task after the Downstack task.

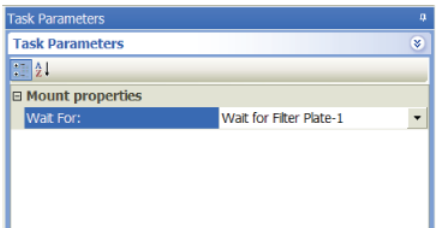


The task parameters for the Wait For task and the Mount task are:

Wait For



Mount



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Dismount task	"Dismount" on page 367
Wait For task	"Wait For" on page 686
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Move to Location (Agilent Bravo)

Description

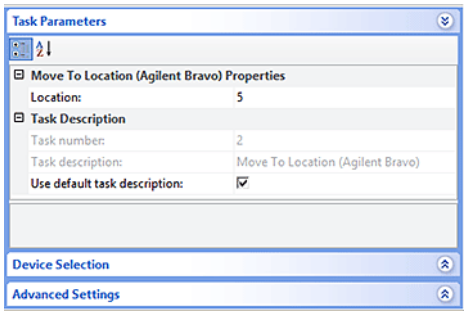
The Move to Location (Agilent Bravo) task (Move To Location (Agilent Bravo)) moves the pipette head to a safe distance above a specified deck location. The safe distance is set in the Bravo profile.

Typically, this task is used in Startup and Cleanup Protocols to move the pipette head out of the way. For example, you can move the pipette head above deck location 5 before placing a microplate at location 1.

Task is available for...	Task is available in...
Bravo Platform	Startup Protocol Main Protocol, Bravo Subprocess Cleanup Protocol

Task parameters

After adding the Move To Location task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location	The deck location the pipette head will move to. The height above the deck location is determined by the *z*-axis safe position setting in the profile.

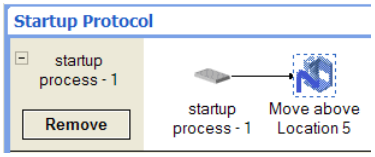
Example: Move the Bravo head before the main protocol starts

Goal

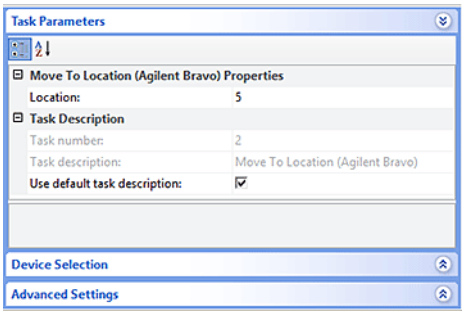
Before the main protocol starts, move the Bravo head to location 5.

Implementation

In the startup protocol, add a Move to Location task. Specify that the Bravo head should move to location 5.



The task parameter for the Move to Location task is:



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Place Plate

Description

The Place Plate task (Place Plate) moves labware to a specified location. If the location has a barcode reader installed, the Place Plate task moves the labware to the location and reads the barcode.

Task is available for...	Task is available in...
Any device	Main Protocol

Requirements

To read barcodes at the specified location such as a platepad, you must:

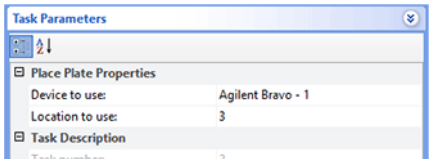
- Set up the barcode reader in the device file.

- Add the device that has the barcode reader, and then specify the location of the barcode reader. For example, if the barcode reader is installed on a platepad, specify the side on which the barcode reader is installed.

If you are placing a labware on a device or accessory, make sure the device or accessory is configured.

Task parameters

After adding the Place Plate task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Device to use	The device where you want to place the labware.
Location to use	The location on the device to place the labware.

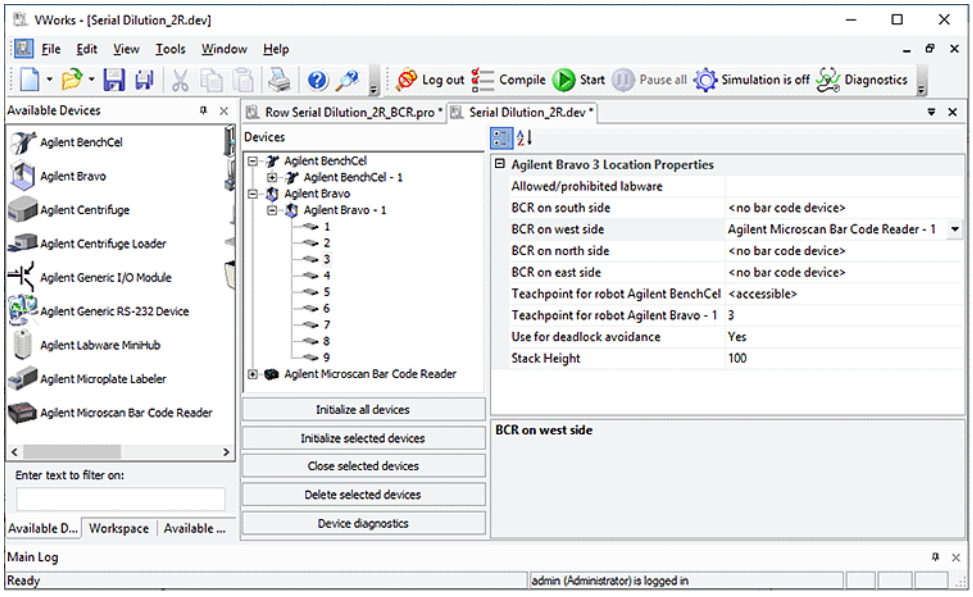
Example: Place a microplate on a Bravo platepad and read its barcode label

Goal

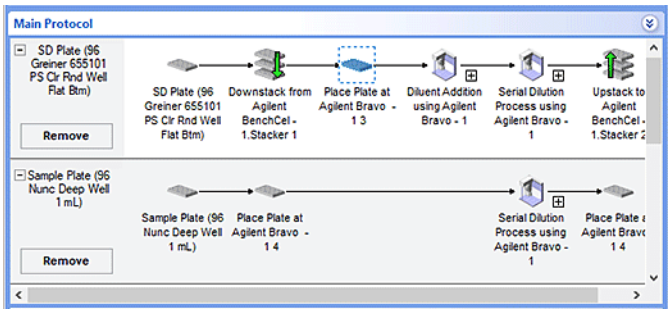
Place the Source Plate at the specified platepad on the Bravo Platform and read the barcode label. The barcode reader is installed on the west side of the platepad.

Implementation

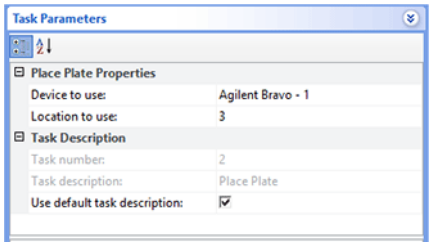
When adding the Bravo device in the device file, specify the barcode reader on the west side of the platepad. In the following example, the barcode reader is on deck location 3.



Add a Place Plate task at the desired point in the protocol.



The task parameters for the Place Plate task are:



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Print and Apply

Description

The Print and Apply task ( Print and Apply) allows you to:

- Specify the contents to be printed on a microplate label.
- Print the specified contents on a label.
- Apply the label to a labware.

Task is available for...	Task is available in...
Microplate Labeler	Main Protocol

Requirements

Before adding the Print and Apply task, make sure you complete the following:

Step	For this task...	See...
1	Set up the Microplate Labeler.	"Setting up the Microplate Labeler" on page 381
2	Specify the label format.	"Specifying the label format" on page 382
3	*Optional*. Create barcode data files.	"Creating barcode data files" on page 382
4	Select the option to lower the plate stage when labeling.	"Lowering the plate stage" on page 383
5	*Optional*. Specify the label location on labware for barcode reading.	"Specifying label location for barcode reading" on page 384

Setting up the Microplate Labeler

For instructions on how to set up the Microplate Labeler (installing the device, adding the device in the VWorks software, creating a profile for the device in diagnostics, and so on), see the *Microplate Barcode Labeler User Guide*.

Specifying the label format

In Microplate Labeler Diagnostics, you specify the label format (label design). The format you specify includes the following:

- Number of fields to be printed (up to six are permitted)
- Type of content to be populated in a field (human-readable text or barcode)
- Font type of the human-readable text, or symbology of the barcode
- Position (coordinates) of the information on the label
- Field attributes, such as field size
- Position (coordinates) of the information on the label

For instructions, see the *Microplate Barcode Labeler User Guide*.

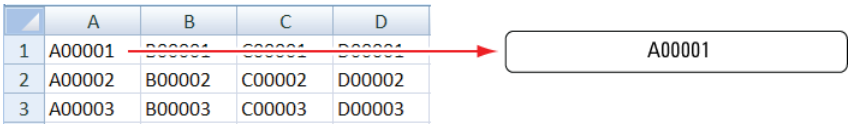
After you specify the label format, you can use the Print and Apply task to specify the content that will populate each field. When you run the protocol that contains the Print and Apply task, the software substitutes the data (label content) for the text and barcode fields in the label format, and then prints the label.

Creating barcode data files

The Print and Apply task provides many methods to create barcode label contents. Two of the methods require the use of barcode data files:

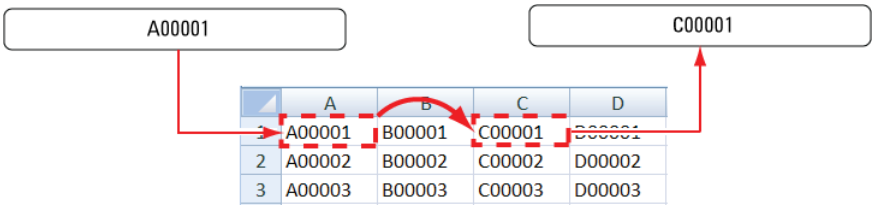
- The software reads data from the data file and prints that data on the label.

  Note: You specify the row and column number of the starting value when you set the Print and Apply task parameter. For instructions, see "Setting the task parameters" on page 386.



- The software reads the barcode on one side of a labware, looks up the barcode (key) in the data file, locates the data found in another column (value), and prints and applies the new label to the same side. The primary use of this function is for label replacement.

  Note: You specify the key and value columns when you set the Print and Apply task parameter. For instructions, see "Setting the task parameters" on page 386.



Barcode data files must meet the following requirements:

- Be a comma-delimited text file with the .csv file name extension
- *Optional.* Contain a header, which can be in any format.
- Contain at least one column. For example, the file can contain four columns, each representing a side of the microplate.

The following example shows a data file displayed in Excel. The file contains four columns, each representing a side of the microplate. In addition, the file contains a header in row 1.

	A	B	C	D
1	North	South	East	West
2	A00001	B00001	C00001	D00001
3	A00002	B00002	C00002	D00002
4	A00003	B00003	C00003	D00003
5				

Ensure that the data file is stored in the barcode file directory that you specify in the Protocol Options area of the protocol.

If you plan to use data files, make sure the files meet the requirements described in this section. In addition, make sure you select the correct data file when setting the Print and Apply task parameters. See "Setting the task parameters" on page 386.

Lowering the plate stage

The Microplate Labeler plate stage has two vertical positions, top and bottom, which are used to accommodate two different sized microplates. By default, the system uses the top position during a protocol run. If the labware has a tall skirt (**1**) or has a raised surface on the sides (**2**), you can specify that the Microplate Labeler use the bottom position (or lower the stage) during the run so that the label can be applied above the skirt or raised surface.

1 **2**



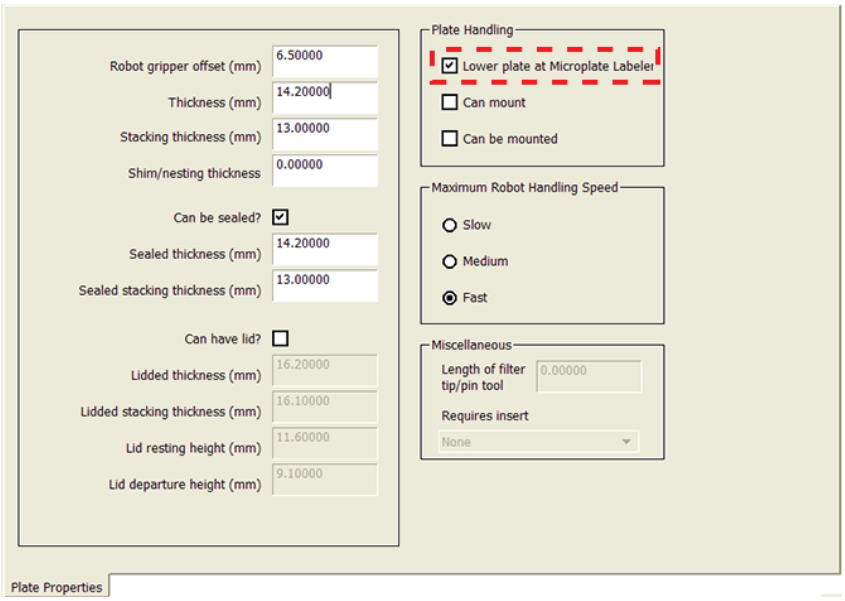
IMPORTANT

Always perform a dry run to verify that the positions are correct for the labware you are processing.

Note: The two vertical plate stage positions are set mechanically. To adjust the positions, see the *Microplate Barcode Labeler User Guide* for instructions.

To lower the plate stage during a run:

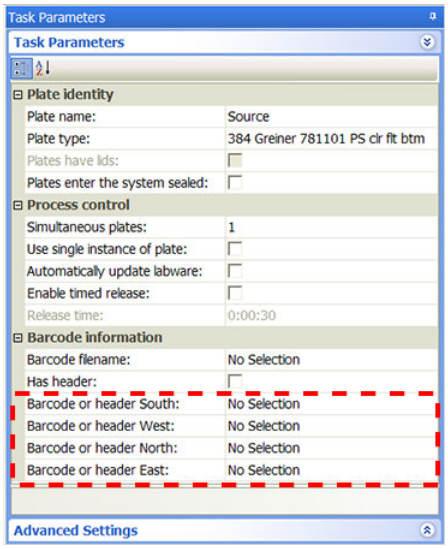
1. Open the **Labware Editor,** and then click the **Plate Properties** tab.
2. Select the labware from the list on the left of the dialog box.
3. In the **Plate Handling** area, select **Lower plate at Microplate Labeler.**



Specifying label location for barcode reading

If you plan to use a barcode reader at the Microplate Labeler, you must specify the location of the barcode label on the labware so that the system will know which side to scan during the run.

You specify the barcode label location when you set the plate parameters. For more information, see "Setting plate parameters" on page 59.



If you plan to scan a new label after it is applied, you must also select the Verify barcodes after Print and Apply option in the Microplate Labeler Diagnostics Profiles tab. For more information, see the *Microplate Barcode Labeler User Guide*.

Selecting devices for the task

After adding the Print and Apply task, you must select a device for the task before you can set the task parameters. If you have multiple devices of the same type, you can:

- Prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.
- Set up a backup pool. If the primary device encounters an error, the software will automatically use the next device in the primary list. However, if all of the devices in the primary list are in an error state, the software will automatically use the device in the backup pool.

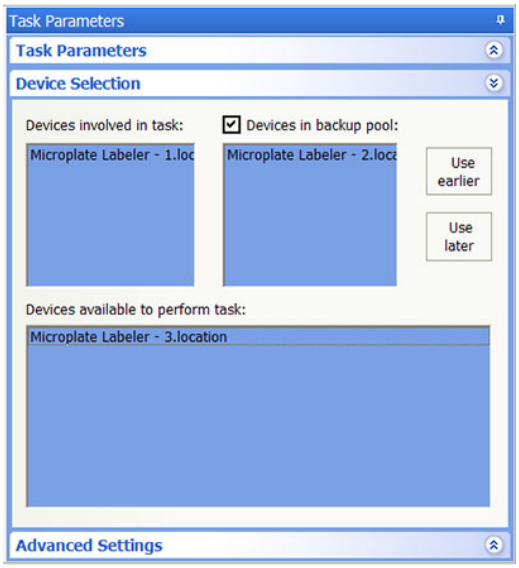
IMPORTANT

The multiple devices must be configured and set up identically. All formats must be identical across the multiple devices. This includes the format names, field names, and definitions.

After adding the Print and Apply task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1. Double-click the desired device in the **Devices available to perform task** area to move it to the **Devices involved in task** area. If you have multiple devices of the same type, you can move them to the **Devices involved in task** area.
2. If you have multiple devices in the **Devices involved in task** area, select a device, and then click **Use earlier** or **Use later** to prioritize it.
3. *Optional.* Select backup devices to use in case all of the devices in the **Devices involved in task** area encounter an error.
   a. Select **Devices in backup pool.**
   b. Drag one or more devices from the **Devices available to perform task** area to the **Devices in backup pool** area.
   c. If you have multiple devices in the backup pool, select a device in the **Devices in backup pool** area, and then click **Use earlier** or **Use later** to prioritize it.



Setting the task parameters

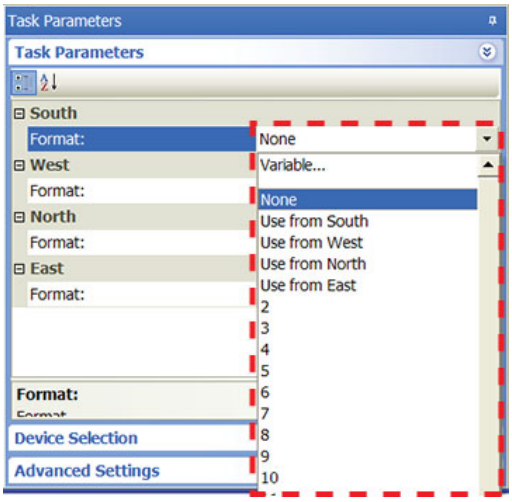
IMPORTANT Make sure the label formats are uploaded to the printer. Be sure to initialize the Microplate Labeler device before you set the task parameters.

After selecting the device to use for the Print and Apply task, you can set the parameters in the **Task Parameters** area. The area lists the four sides of a microplate (south, west, north, and east). For each side, you can select a label format and specify the data that will substitute for the text and barcode fields in the label format.

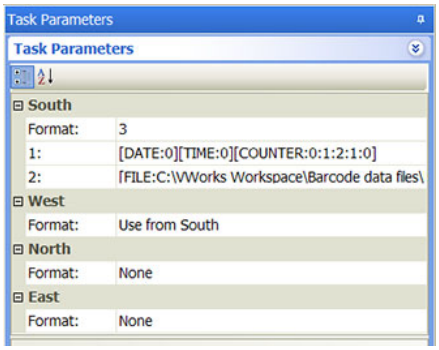
CAUTION **Format selection and field information are saved with the protocol. If the formats on the printer are changed, initializing the device will overwrite the information in the protocol. For example, suppose you created a protocol and selected a format called MyFormat. Later, MyFormat was deleted from the printer. The next time you initialize the device and open the protocol, MyFormat will not appear in the protocol.**

To set the task parameters:

1 For the side on which the barcode label will be applied, select the barcode format in the **Format** list:

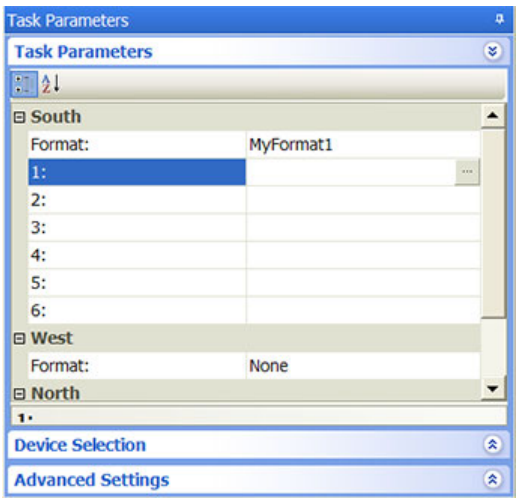


Format selection	Description
None	Indicates no label will be applied.

Format selection	Description
Use from <*side*>	Uses the format from another side of the microplate to print a new label and apply it to the selected side. For example, suppose you have already selected a format and specified the label contents for the South side. You want to use the same format and contents on the West side without having to reselect and respecify the same information. Then in the West Format list, select Use from South. 
Format name or number	Uses a format that was set up in Microplate Labeler Diagnostics. *Note:* If you do not see a list of formats, make sure the label formats are not empty (formats must contain at least one field), the formats are uploaded to the printer, and the Microplate Labeler device is initialized.

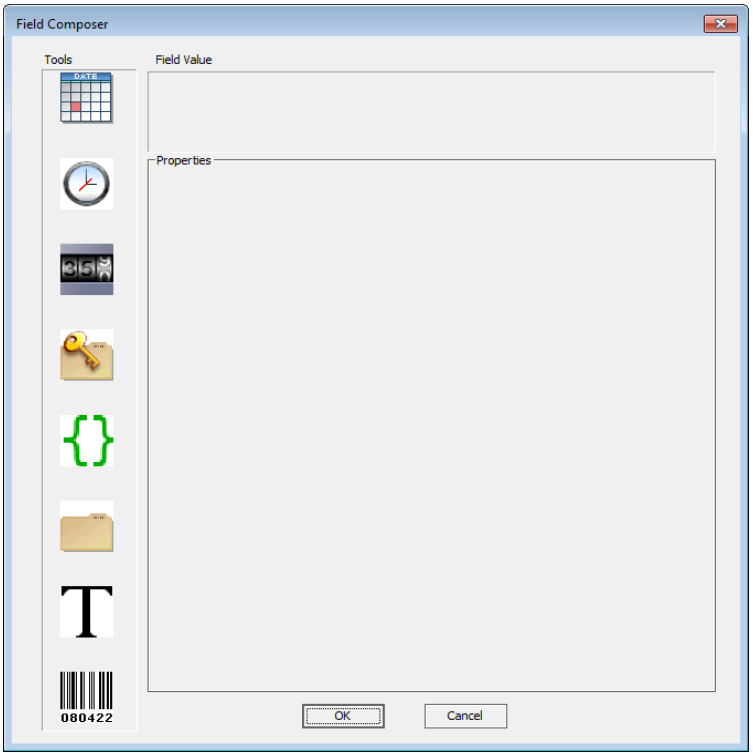
As soon as you select a format, fields appear in the Task Parameters area. You can specify the information you want to print in these fields.

Note: The number of fields that appear depends on the format you select.



2 Click a field, and then click the ··· button that appears. The Field Composer dialog box opens.

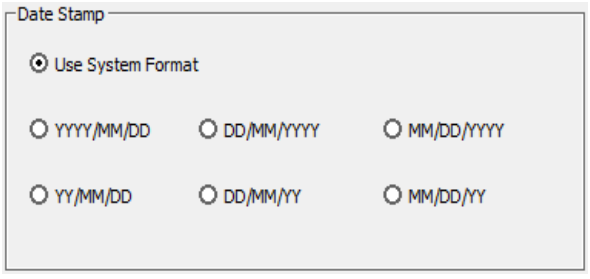
The Field Composer allows you to specify the information to print on the barcode label. For example, you can print the current date and time.

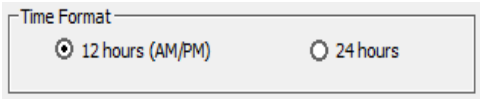
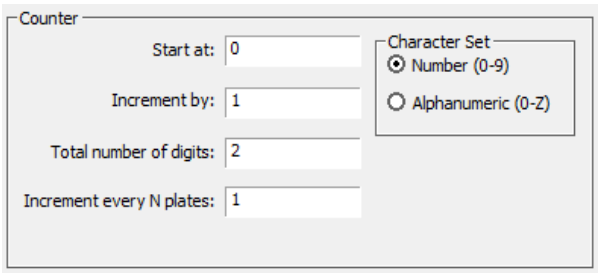


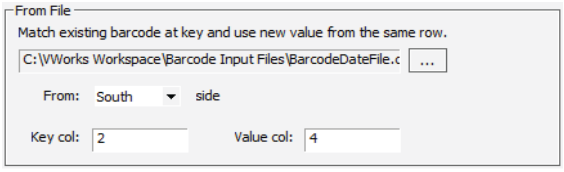
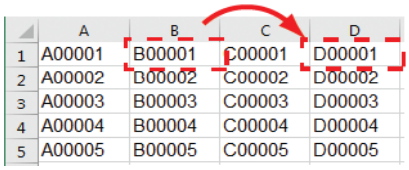
3 In the **Tools** area, double-click one or more of the following icons to specify the information to be printed on the barcode label. The selected icon appears in the **Field Value** area.

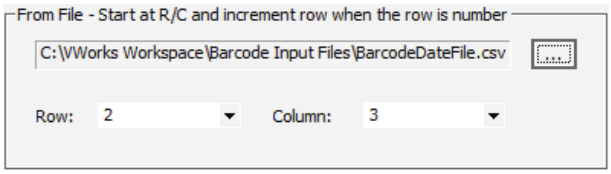
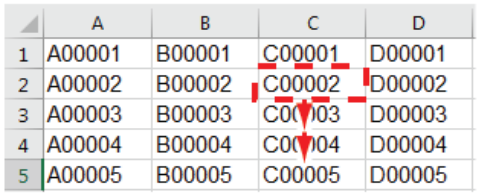
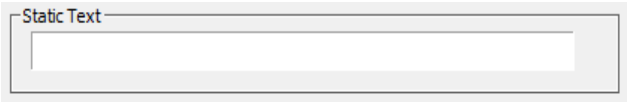
IMPORTANT

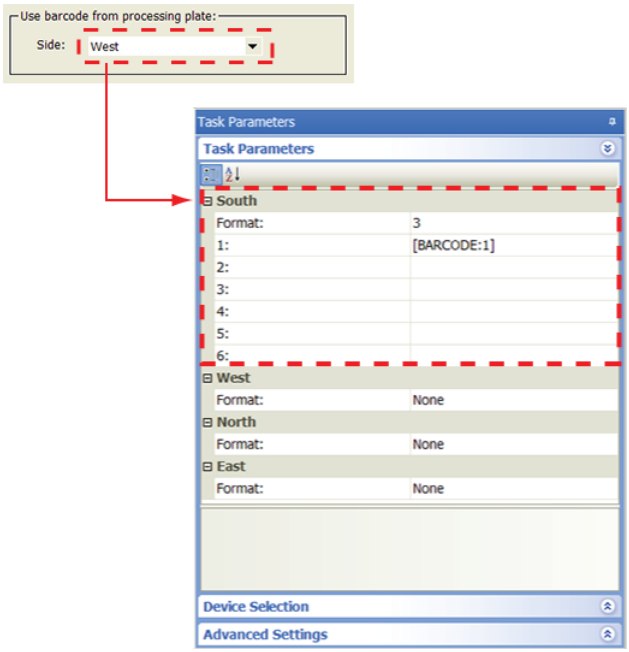
For field limitations, such as the maximum number of characters permitted or symbology-dependent limitations, check the format you set up in Microplate Labeler Diagnostics. See also the *Microplate Barcode Labeler User Guide*.

Icon	Description
	Prints the current date. Click the icon in the **Field Value** area. In the **Properties** area, select the desired date format. YYYY is the year, MM is the month, and DD is the day. *Note:* The Use System Format option uses the local computer's date format. 

Icon	Description
	Prints the current time. Click the icon in the **Field Value** area. In the **Properties** area, select the desired data format: **12 hours (AM/PM)** or **24 hours.** 
	Prints a numeric or alphanumeric value that can be incremented. Click the icon in the **Field Value** area. Set the following in the **Properties** area: • *Character Set*. The option to use either numeric or alphanumeric characters. • *Start at*. The starting value. • *Increment by*. The amount by which the value increments. • *Total number of digits*. The total number of digits or characters, including leading 0s. • *Increment every N plates*. The increment value. For example, 1 increments the value every microplate. 

Icon	Description
	Looks up a value (key) in the specified barcode data file and prints the corresponding value found in the file. Click the icon in the **Field Value** area. In the **Properties** area, click the ... button to select the barcode data file. In the From list, select the scanned side of the labware. Specify the column number that contains the lookup value (key) and the column number that contains the values you want to print. For example, From is set at South, Key col is set at 2, and Value col is set at 4. During the run, a B00001 barcode is scanned on the south side of the labware. The software looks up B00001 in the specified file, and prints the barcode, D00001, found in the fourth column. *Note:* The plate parameters must specify **Barcode or header South** in the Barcode information area so the system knows to scan the south side of the incoming microplates.  

Icon	Description
	Prints the values in the specified barcode input file. Click the icon in the **Field Value** area. In the **Properties** area, click the ... button to select the input file. Specify the row and column number of the starting value. During the run, the software automatically increments to the next row to print the next value. For example, Row is set at 2, Column is set at 3. During the run, the software starts with the value in row 2 column 3 (C00002). Then, the software moves to row 3 column 3 (C00003), row 4 column 3 (C00004), row 5 column 3 (C00005), and so on.  
	Prints the text you specify. Click the icon in the **Field Value** area. In the **Properties** area, type the text you want to print on every microplate. 

Icon	Description
	Copies the barcode data from an existing label on the selected side to print a new label. *Note:* You can only copy barcode data from another label. You cannot copy human-readable text. Click the icon in the **Field Value** area. In the **Properties** area, select the side (barcode) to be copied. For example, an incoming microplate already has a label on the west side. You want to copy the barcode data from the west-side label to print a new label, and apply the new label to the south side. In the Task Parameters area, you specify that the south side of the labware will be labeled. In the Field Composer dialog box, you specify Use barcode from processing plate, and select West. 

When you are finished, click **OK.** The information you specified appears in the Task Parameters area.

Repeat the procedure if you want to print labels on other sides of the microplate.

Example 1: Copy label contents from one side of the microplate and print and apply to a new side

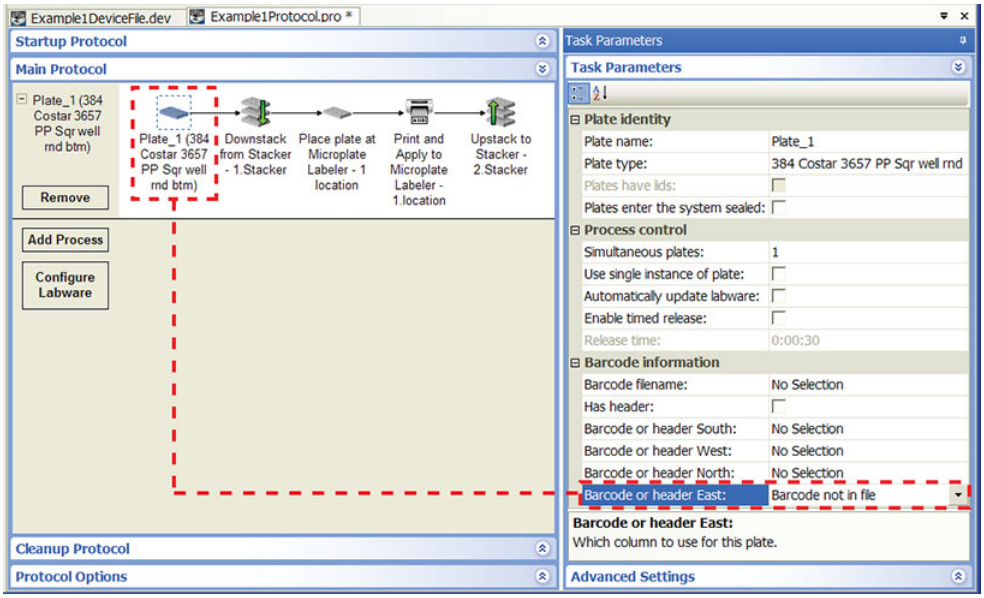
Goal

For each incoming microplate, scan the label on the east side of the microplate, print the same contents on a new label, and apply the new label to the north side of the microplate.

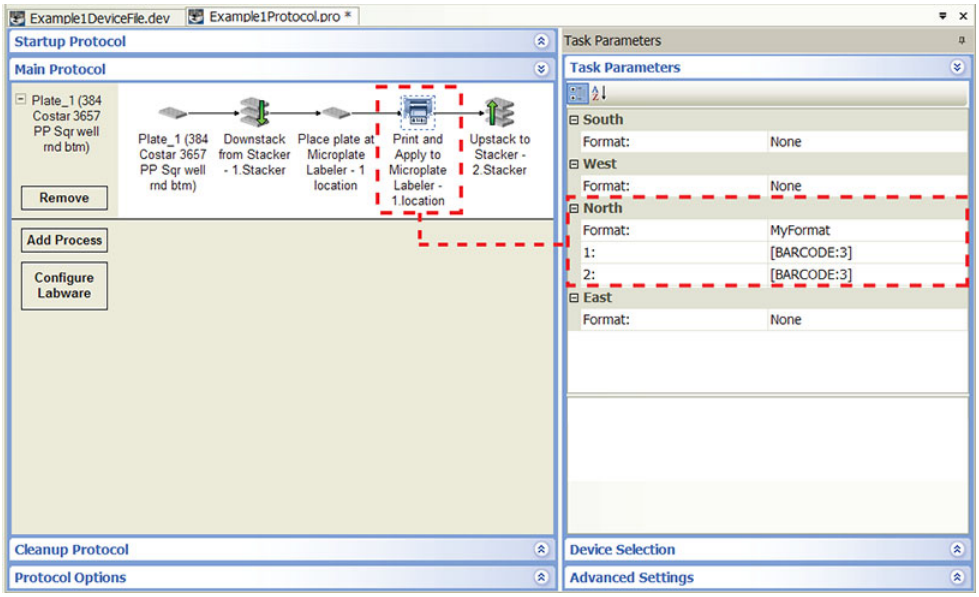
Implementation

Note: This example assumes that the Microplate Labeler is set up correctly and the format, MyFormat, is already defined and loaded to the printer. MyFormat contains two fields. Field 1 is a human-readable text field. Field 2 is a barcode field. The device file and protocol are correctly created.

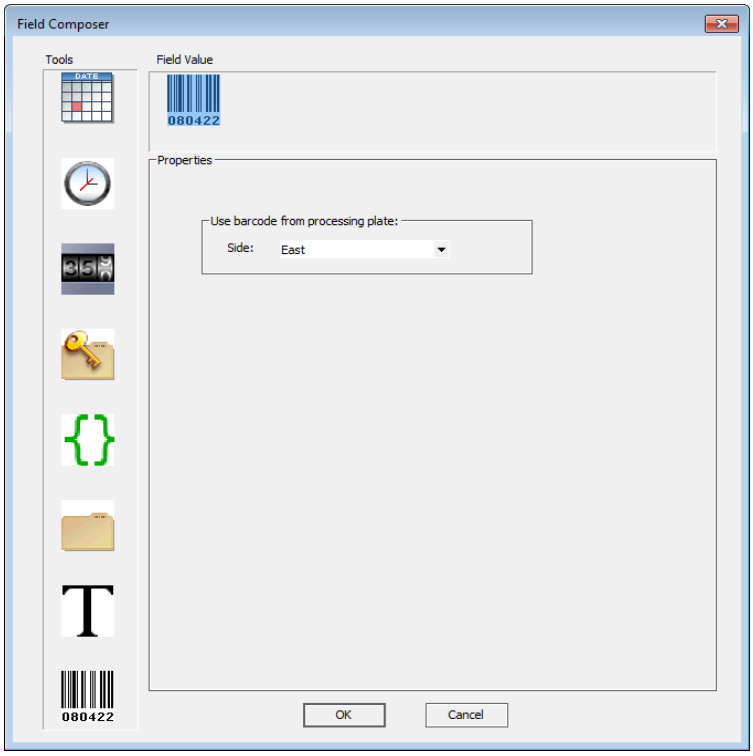
When setting the plate parameters, select **Barcode or header East** in the **Barcode information** area so the system knows to scan the east side of the incoming microplates. No barcode verification file will be used, so select **Barcode not in file.**



When setting the Print and Apply task parameter, select MyFormat for the side you want to label: **North.**



For each of the two format fields (1 and 2), open the Field Composer dialog box and double-click the barcode icon in the **Tools** area to add it to the Field Value area. Because you want to copy the barcode information from the east side of the microplate, select **East** from the **Side** list.



Example 2: Use a barcode data file to print and apply labels

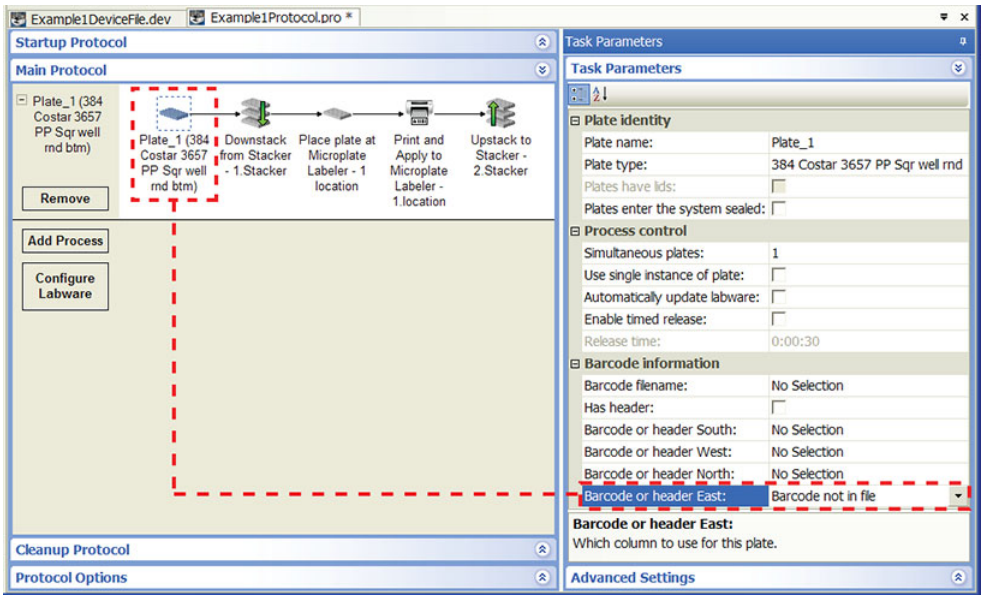
Goal

For each incoming microplate, scan the label on the east side of the microplate, locate the information (key) in the first column in the data file, print the corresponding information (value) from the third column in the file on a new label, and apply the new label to the same (east) side of the microplate.

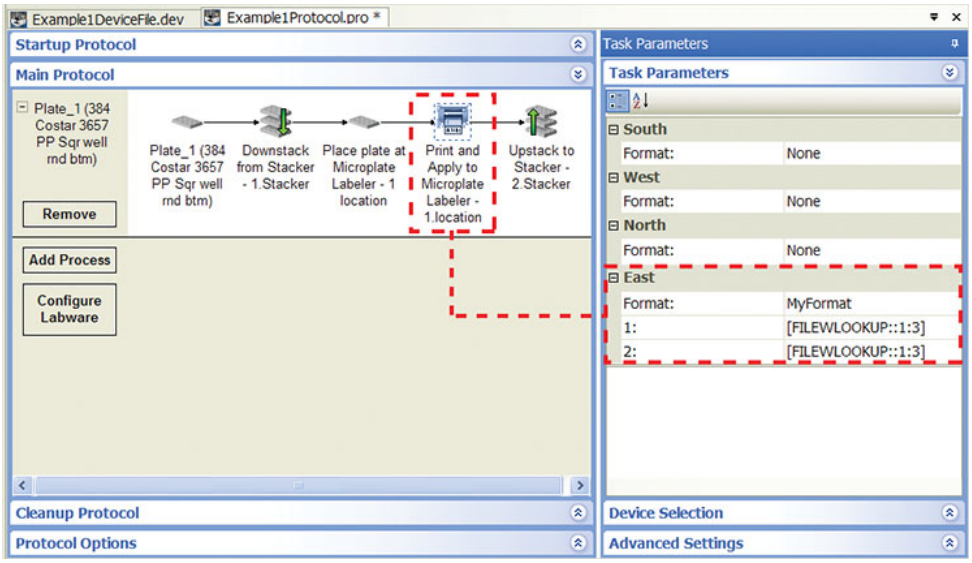
Implementation

Note: This example assumes that the Microplate Labeler is set up correctly and the format, MyFormat, is already defined and loaded to the printer. MyFormat contains two fields. Field 1 is a human-readable text field. Field 2 is a barcode field. The barcode data file is stored in C:\VWorks Workspace\Barcode data files. The device file and protocol are correctly created.

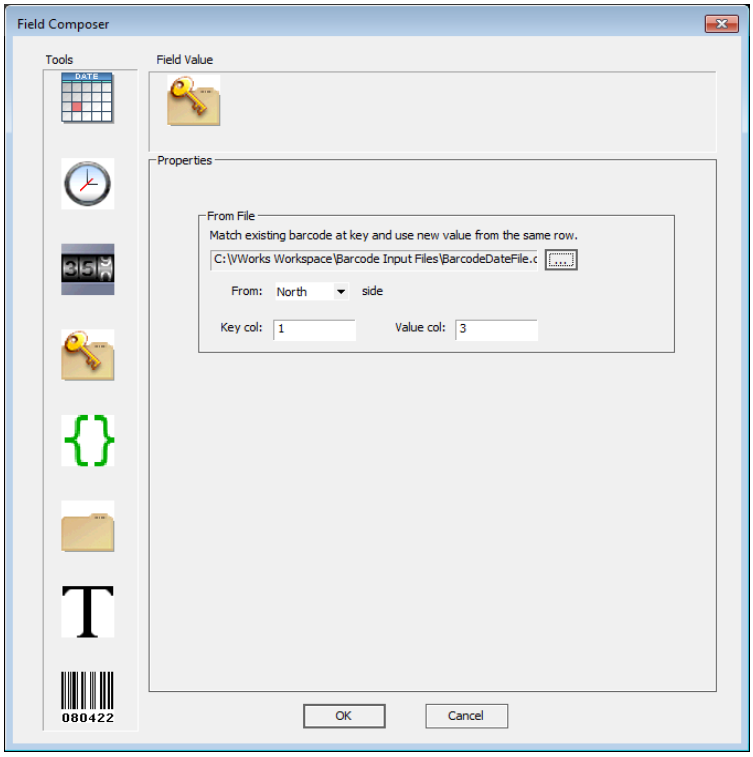
When setting the plate parameters, select **Barcode or header East** in the **Barcode information** area so the system knows to scan the east side of the incoming microplates. No barcode verification file will be used, so select **Barcode not in file.**



When setting the Print and Apply task parameter, select MyFormat for the same side: **East.**



For each of the two format fields, open the Field Composer dialog box and double-click the key-file icon in the **Tools** area to add it to the Field Value area. In the **From File** area, locate and select the data file to use. In the **From** list, select the East side. In the **Key col** box, type 1. In the **Value col** box, type 3.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • *Microplate Barcode Labeler User Guide*
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Barcode data file and barcode input file	"Tracking barcodes" on page 90
Protocol options	"Setting protocol options" on page 42
Rotate Stage (Microplate Labeler) task	"Rotate Stage (Agilent Microplate Labeler)" on page 403
Writing JavaScript for the Print and Apply task	"Using JavaScript with the CentrifugeAuto task" on page 136
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Relid

Description

The Relid task (

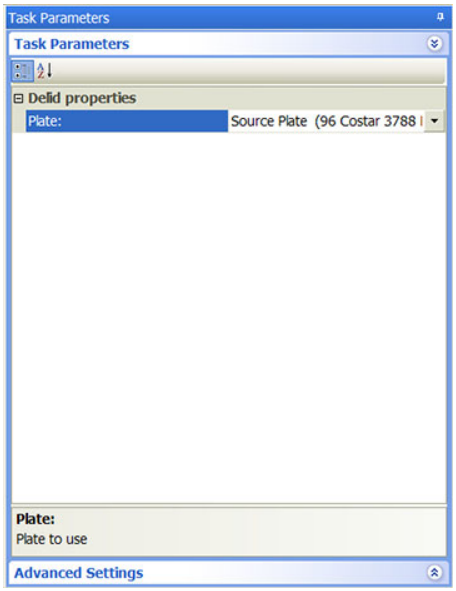
) places lids on the microplates.

Task is available for...	Task is available in...
BenchCel Microplate Handler	Startup Protocol Main Protocol Cleanup Protocol
Bravo Platform	Startup Protocol Main Protocol Cleanup Protocol

Task Parameters

Bravo Platform

When relidding microplates on the Bravo Platform, you add the Relid task within the Bravo Subprocess. After adding the Relid task at the desired point in the Bravo Subprocess, select the microplate you want to relid in the **Task Parameters** area, as the following example shows.



Parameter	Description
Plate	The microplate that will be relidded.

Device selection

You do not need to select a device for the Relid task. The system automatically uses the correct device to relid the microplates.

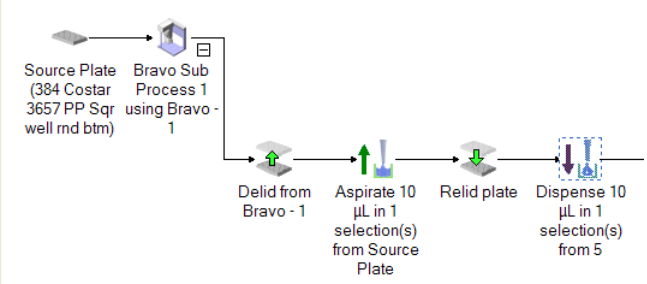
Example: Delid and relid microplates

Goal

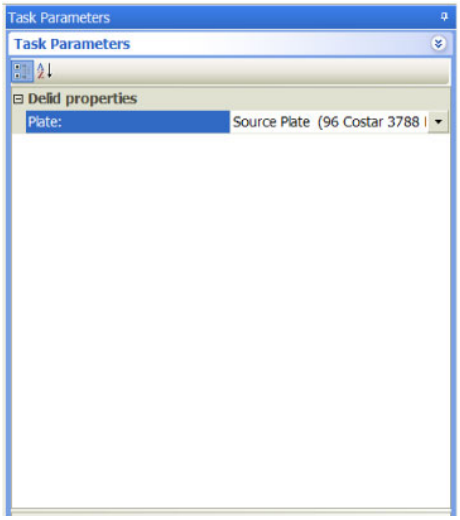
On the Bravo Platform, remove a lid from the Source Plate for an Aspirate task. After the task is finished, relid the microplate.

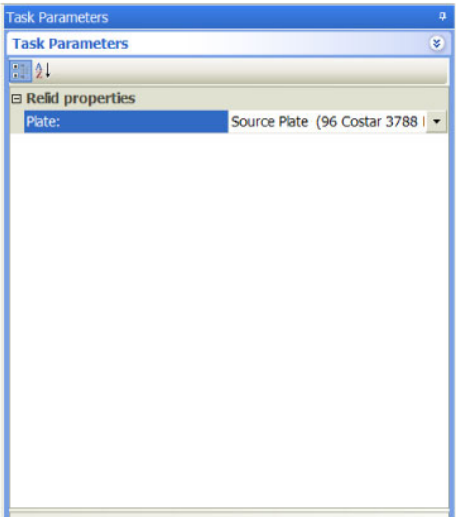
Implementation

Within the Bravo Subprocess, add a Delid task before the Aspirate task to remove the lid. Add a Relid task after the Aspirate task to place the lid back on the microplate.

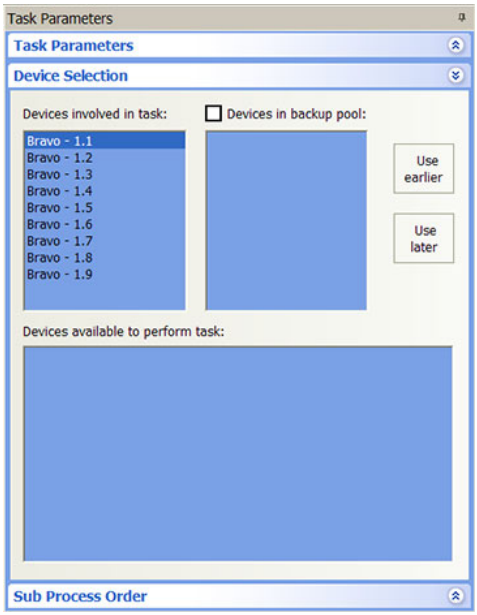


The task parameters for the Delid and Relid tasks are:





The device selection for the Delid task is as follows. Note that you do not need to select a device for the Relid task.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Delid task	"Delid" on page 363
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Reserve Location

Description

The Reserve Location task (Reserve Location) reserves a location on the Bravo Platform for a specified length of time. The task is typically used to incubate a microplate at that location.

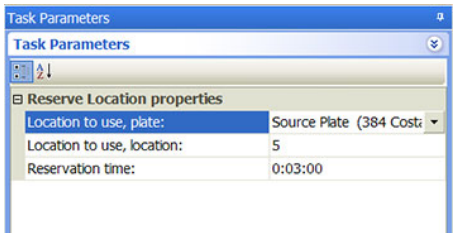
Note:

- To incubate labware on a platepad, use the Incubate task.
- To incubate a labware in a storage device such as a Labware MiniHub, use the Storage Incubate task.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Task parameters

After adding the Reserve Location task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location to use, plate	The labware that will be placed at the specified location during the reservation period.
Location to use, location	The location that you want to reserve. <auto-select> automatically places the labware at the first-available or appropriate location for the task. If accessories are installed on the deck, the software uses the accessory configuration information in Bravo Diagnostics to determine the correct location for the task.
Reservation time	The length of time that the location is reserved. The time starts when the desired labware arrives at the location.

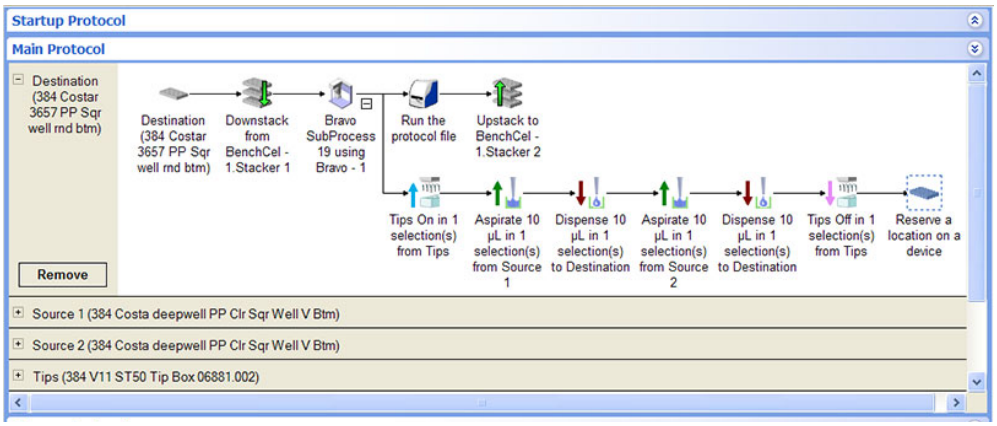
Example: Reserving a deck location for incubation

Goal

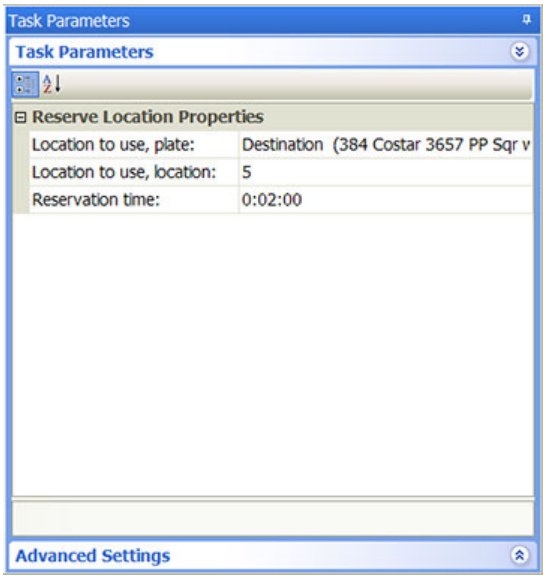
Aspirate contents from two source microplates and dispense into a destination microplate. Incubate the destination microplate at Bravo deck location 5 for 2 minutes before reading the microplate.

Implementation

In the protocol, add a Reserve Location task after the liquid-handling tasks.



The task parameters for the Reserve Location task are shown in the following example.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67

For information about...	See...
Incubate task	"Incubate" on page 370
Storage Incubate task	"Storage Incubate" on page 434
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Rotate Stage (Agilent Microplate Labeler)

Description

The Rotate Stage (Agilent Microplate Labeler) task ( Rotate Stage (Agilent Microplate Labeler)) rotates the Microplate Labeler stage 180°. You use this task to change the orientation of microplates in preparation for label applications.

Task is available for...	Task is available in...
Microplate Labeler	Main Protocol

Task parameters

The Rotate Stage (Agilent Microplate Labeler) task does not have task parameters.

Device selection

You must select a device for Rotate Stage (Agilent Microplate Labeler) tasks. If you have multiple devices of the same type, you can:

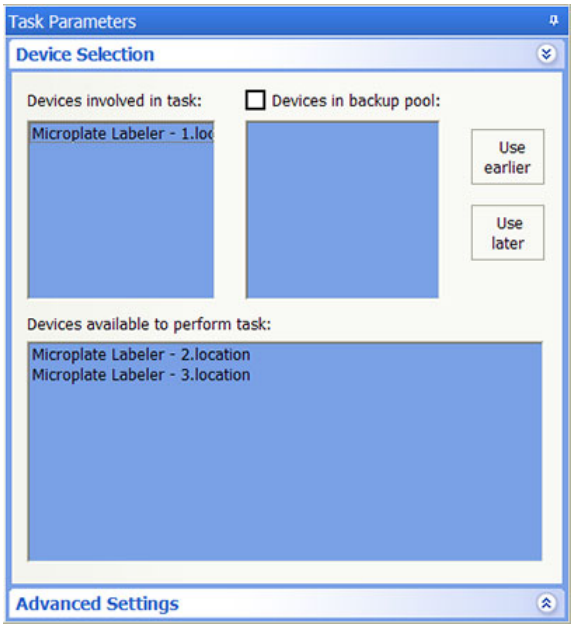
- Prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.
- Set up a backup pool. If the primary device encounters an error, the software will automatically use the next device in the list. However, if all of the devices in the list are in an error state, the software will automatically use the device in the backup pool.

IMPORTANT The multiple devices must have the same setup and configuration.

After adding the Rotate Stage (Agilent Microplate Labeler) task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1. Double-click a device in the **Devices available to perform task** area to add it to the **Devices involved in task** area.
2. To prioritize its use, select the device in the **Devices involved in task** area, and then click **User earlier** or **Use later.** The devices (deck locations) that are higher in the list are favored by the software scheduler during the protocol run.
3. *Optional.* Select backup devices to use in case all of the devices in the **Devices involved in task** area encounter an error.
   a. Select **Devices in backup pool.**
   b. Drag one or more devices from the **Devices available to perform task** area to the **Devices in backup pool** area.
   c. If you have multiple devices in the backup pool, select a device in the **Devices in backup pool** area, and then click **Use earlier** or **Use later** to prioritize it.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • *Microplate Barcode Labeler User Guide*
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Print and Apply task	"Print and Apply" on page 381
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343

For information about...	See...
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Seal (Agilent PlateLoc)

Description

The Seal (Agilent PlateLoc) task ( Seal (Agilent PlateLoc)) seals microplates using the PlateLoc Sealer.

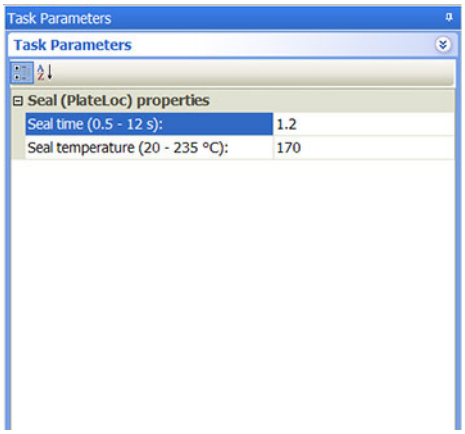
Note:

- If you require different sealing temperatures to accommodate different microplate types, Agilent recommends that you use a different PlateLoc Sealer for each sealing temperature. Using different PlateLoc Sealers that have dedicated temperature settings prevents time lost as the device heats or cools between microplate types.
- The PlateLoc Sealer immediately starts adjusting to the **Startup seal temp** defined in its profile when the profile is initialized. Initialization occurs when the protocol is first run or when you click **Initialize this profile** in the diagnostics dialog box.

Task is available for...	Task is available in...
PlateLoc Sealer	Main Protocol

Task parameters

After adding the Seal (PlateLoc) task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Seal time	The length of time the hot plate is in contact with the seal material and the microplate.
Seal temperature	The temperature of the hot plate when the seal cycle starts.

Device selection

You must select a device for Seal (Agilent PlateLoc) tasks. If you have multiple devices of the same type, you can:

- Prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.
- Set up a backup pool. If the primary device encounters an error, the software will automatically use the next device in the list. However, if all of the devices in the list are in an error state, the software will automatically use the device in the backup pool.

IMPORTANT

The multiple devices must have the same setup and configuration.

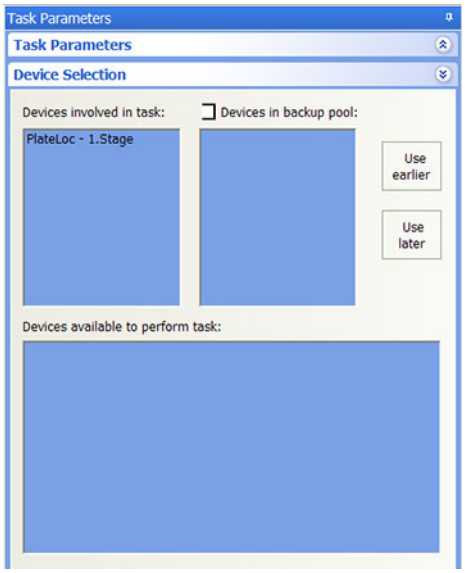
After adding the Seal (Agilent PlateLoc) task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1. Double-click a device in the **Devices available to perform task** area to add it to the **Devices involved in task** area.
2. To prioritize its use, select the device in the **Devices involved in task** area, and then click **User earlier** or **Use later.** The devices that are higher in the list are favored by the software scheduler during the protocol run.
3. *Optional.* Select backup devices to use in case all of the devices in the **Devices involved in task** area encounter an error.
   a. Select **Devices in backup pool.**

b Drag one or more devices from the **Devices available to perform task** area to the **Devices in backup pool** area.

c If you have multiple devices in the backup pool, select a device in the **Devices in backup pool** area, and then click **Use earlier** or **Use later** to prioritize it.



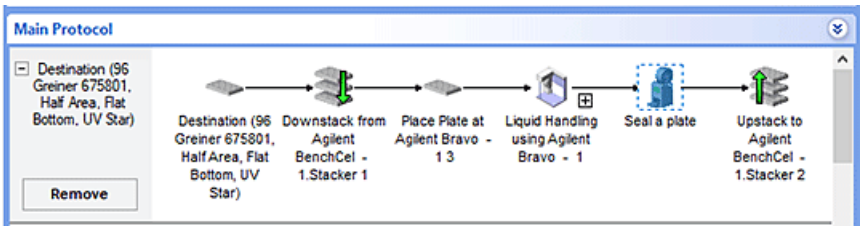
Example

Goal

Downstack microplates from a BenchCel Microplate Handler for liquid-handling tasks on the Bravo Platform, seal them, and then upstack them back to the BenchCel Microplate Handler.

Implementation

Add a Seal (Agilent PlateLoc) task after the Bravo Platform Subprocess.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Downstack task	"Mount" on page 374

For information about...	See...
Upstack task	"Upstack" on page 443
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Waste

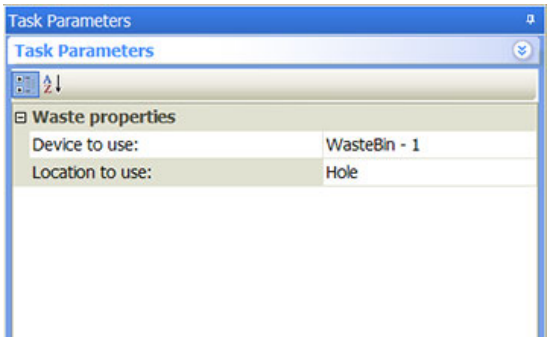
Description

If your automation system includes a trash container or hole, you can use the Waste task () to move a lid or labware to the trash container or hole.

Task is available for...	Task is available in...
Waste Bin	Main Protocol

Task parameters

After adding the Waste task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Device to use	The waste bin device you want to use for the task.
Location to use	The waste bin location.

Example: Discard filter microplates after filtering process

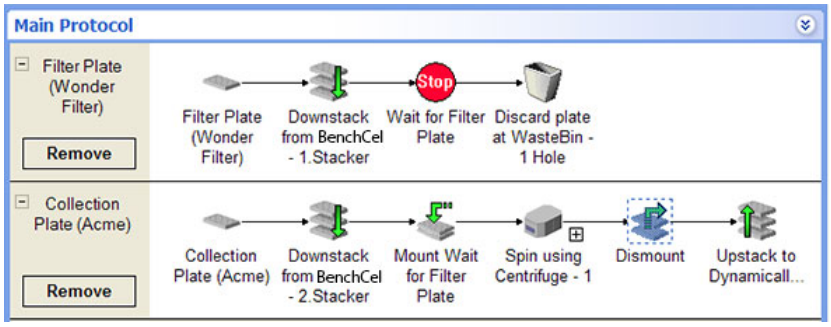
Goal

Downstack a reservoir (collection microplate) from a BenchCel stacker and place it on a platepad. Downstack a filter microplate from another BenchCel stacker and place it on top of the collection microplate. The assembly (filter microplate with collection microplate) is centrifuged. Remove the filter microplate from the collection microplate and discard it in the waste bin. Upstack the collection microplate containing the elution for future processing.

Implementation

Create one process for the collection microplate (Collection Plate) and another process for the filter microplate (Filter Plate). See "Example: Dismount a filter microplate from a collection microplate" on page 368 for details about the Mount, Dismount, and Wait For tasks.

In the Filter Plate process, add a Waste task after the Wait For task. The Waste task discards the filter microplate after the Centrifuge task is finished.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Downstack task	"Mount" on page 374
Upstack task	"Upstack" on page 443
Mount task	"Mount" on page 374
Dismount task	"Dismount" on page 367
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

8 Setting parameters for microplate storage tasks

This chapter contains the following topics:

- "clearInventory" on page 412
- "Downstack" on page 414
- "exportDatabase" on page 418
- "importCsvToInventory" on page 420
- "Load" on page 422
- "Reorder" on page 427
- "Scan Stack" on page 432
- "Storage Incubate" on page 434
- "Unload" on page 438
- "Upstack" on page 443

clearInventory

Description

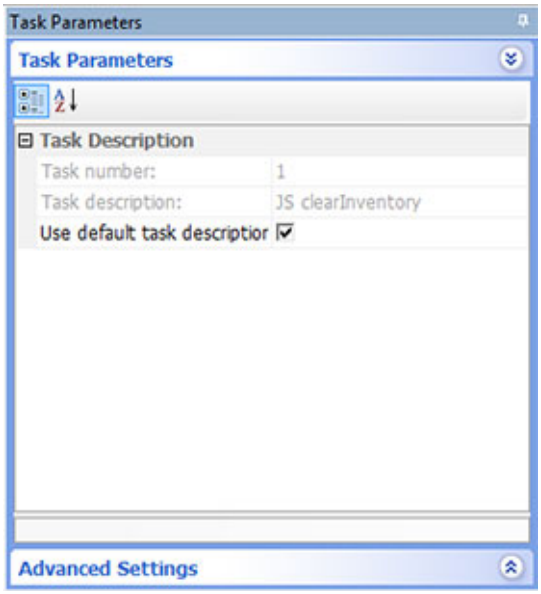
The clearInventory task (JS clearInventory) deletes all labware data from the inventory database. Use this task to clear the inventory at the desired point in the protocol.

If you want to automatically clear the inventory every time the run starts, select **Clear inventory** in Protocol Options. See "Setting protocol options" on page 42.

Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

Task parameters

After adding the clearInventory task at the desired point in the protocol, the following Task Parameters are set by default. You can edit the Task description as desired.



Parameter	Description
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.

Parameter	Description
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Process plate properties	"Setting plate parameters" on page 59
exportDatabase task	"exportDatabase" on page 418
importCsvToInventory task	"importCsvToInventory" on page 420
Load task	"Load" on page 422
Unload task	"Unload" on page 438
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

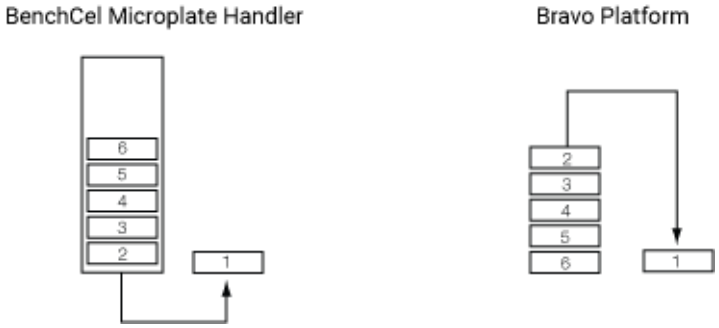
Downstack

Description

The Downstack task (

) moves labware as follows:

- Out of the bottom of stacking devices such as the BenchCel Microplate Handler
- Off of the top of a stack of labware at a Bravo deck location



For example, you can use the Downstack task to move microplates out of a BenchCel stacker and onto the plate stage of a device.

You can use a single Downstack task to move labware out of multiple stacking devices. For example, after all the microplates are moved out of one stacker, the Downstack task can continue at a second stacking device.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess
BenchCel Microplate Handler	Main Protocol

Requirements

Bravo Platform

To use the Downstack task in a Bravo Subprocess, you must:

- Specify the maximum stack height in the Bravo deck location properties area.
- Add a Scan Stack task for each Downstack task.

BenchCel Microplate Handler

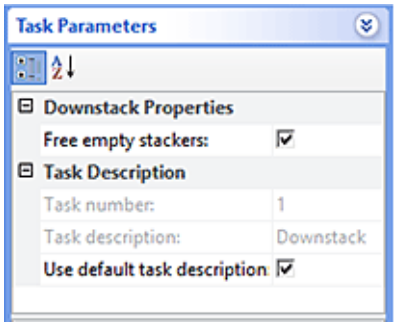
The software always determines a location for a labware before it is downstacked from the BenchCel stacker. If you add a subprocess immediately after a Downstack task, the system will downstack the labware and place it at the location where the first subprocess task will be performed.

If you want to quarantine microplates with the wrong orientation immediately after they are downstacked from a BenchCel stacker, add a Place Plate task immediately after the Downstack task. Doing so ensures that the labware will be quarantined before proceeding to downstream tasks.

For information on how to set up quarantine criteria, see "Setting up automated error responses" on page 735.

Task parameters

After adding the Downstack task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Free empty stackers	The option to allow this stacker, when emptied, to become available for Upstack or Reorder tasks.
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Device selection

You must select a device for Downstack tasks. If you have multiple devices of the same type, you can prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.

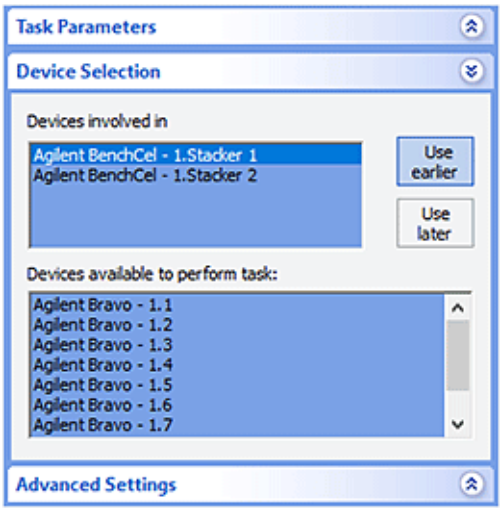
IMPORTANT

The multiple devices must have the same setup and configuration.

After adding the Downstack task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1. Double-click a device or deck location in the **Devices available to perform task** area to add it to the **Devices involved in task** area.
2. To prioritize its use, select the device in the **Devices involved in task** area, and then click **User earlier** or **Use later.** The devices that are higher in the list are favored by the software scheduler during the protocol run.



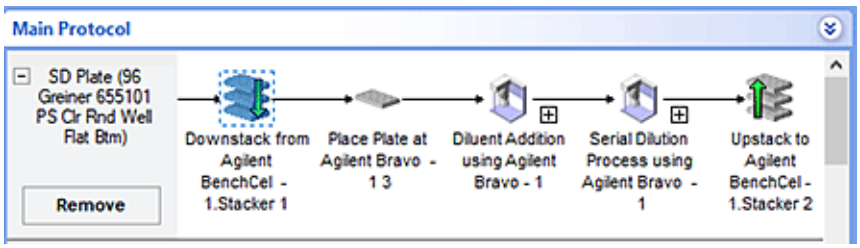
Example: Downstack microplates from BenchCel stacker

Goal

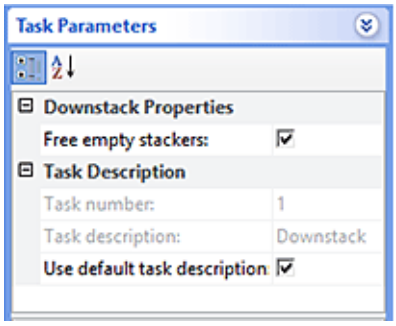
Downstack the Destination Plate from BenchCel 1 stacker for liquid-handling tasks at the Bravo Platform. After the stack is emptied, the stack can be used for Upstack and Reorder tasks.

Implementation

Add a Downstack task before the Bravo Subprocess task that contains the liquid-handling subroutine.



The Free empty stackers parameter is selected and device selection is Agilent BenchCel-1.Stacker 1:



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Scan Stack task	"Scan Stack" on page 432
Upstack task	"Upstack" on page 443
Reorder task	"Reorder" on page 427
Quarantining labware	• "Setting plate parameters" on page 59 • "Setting up automated error responses" on page 735
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

exportDatabase

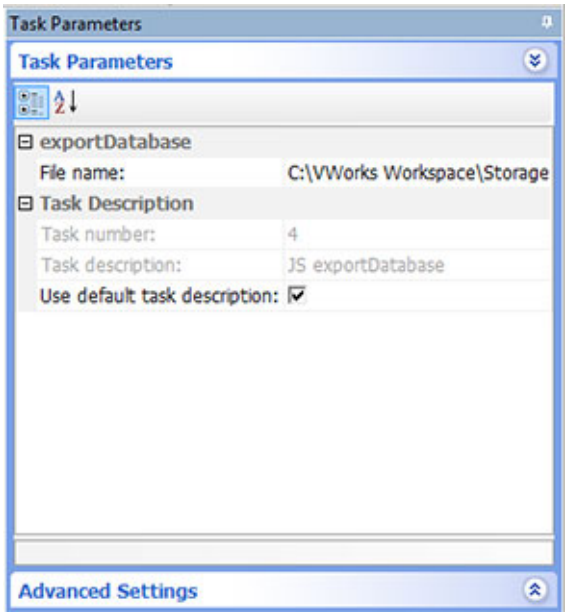
Description

The exportDatabase task (JS exportDatabase) exports all data in the inventory database to the specified location. You can use this task to create a backup copy of the inventory database at the desired point in the protocol.

Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

Task parameters

After adding the exportDatabase task at the desired point in the protocol, set the following **Task Parameters**:



Parameter	Description
File name	The path and name of the inventory file to be exported. Click the browse button to select the location and provide a name for the exported file.
Task number	The number that indicates the position of the task in the protocol.

Parameter	Description
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
clearInventory task	"clearInventory" on page 412
importCsvToInventory task	"importCsvToInventory" on page 420
Scan Stack task	"Scan Stack" on page 432
Upstack task	"Upstack" on page 443
Reorder task	"Reorder" on page 427
Quarantining labware	• "Setting plate parameters" on page 59 • "Setting up automated error responses" on page 735
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

importCsvToInventory

Description

The importCsvToInventory task (

) imports labware data in the specified CSV file into the inventory database.

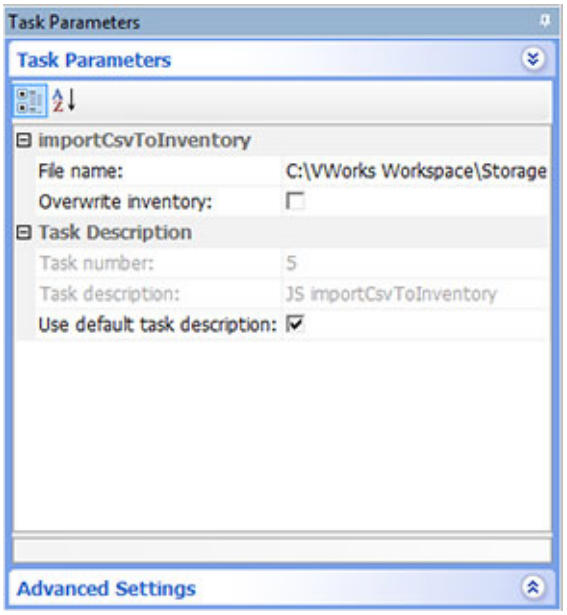
Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

For the list of CSV requirements, see the *VWorks Automation Control Setup Guide*

You can use this task to add a large amount of labware data in the inventory database. For example, in the protocol, you can add a Load task to fill a storage device with labware. If the labware do not have barcodes, or if your system does not have a barcode reader, you can add the importCsvToInventory task to enter the corresponding labware data in the inventory database.

Task parameters

After adding the importCsvToInventory task at the desired point in the protocol, set the following **Task Parameters**:



Parameter	Description
File name	The path and name of the CSV file. Click the field and then click the browse button to locate and select the CSV file.
Overwrite inventory	The option to replace the existing entry in the database with the data in the CSV file.
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
clearInventory task	"clearInventory" on page 412
exportDatabase task	"exportDatabase" on page 418
Scan Stack task	"Scan Stack" on page 432
Upstack task	"Upstack" on page 443
Reorder task	"Reorder" on page 427
Quarantining labware	• "Setting plate parameters" on page 59 • "Setting up automated error responses" on page 735
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Load

Description

The Load task () instructs a robot to move a defined set of labware into a storage device.

Note: If you want to move a defined set of labware from one storage device to another, use the Unload task and Load task in sequence. For information about the Unload task, see "Unload" on page 438.

Task is available for...	Task is available in...
Labware MiniHub	Main Protocol

Storage location selections

When you add the Load task, you can select the storage locations in one of the following ways:

- *Native locations*. Native locations are known locations in a device file. For example, if the Labware MiniHub is in the device file, you can select a cassette in that carousel for the Load task. When you select a cassette, all 16 slots in the cassette are available for the task. If you have only 10 microplates to load into that cassette, six of the slots will remain empty.
- *Location groups*. Location groups are device locations that are grouped together. For example, a location group might consist of 12 Labware MiniHub slots as follows: cassette 1 (5 slots), cassette 2 (5 slots), and cassette 3 (2 slots). If you want to load exactly 12 microplates, you can select that particular location group.

Labware loading order

Under some circumstances, the original order of the labware in plate groups might not be maintained during the Load tasks:

- If the **Dynamically assign empty slot to load to storage device** option is selected in the Protocol Options, the software will find an empty slot to load labware if the original slot is occupied.
- The software always unloads the first labware in the plate group. For example, if a plate group contains microplates 1 through 5, microplate 1 is unloaded, then microplate 2, and so on. If microplate 1 is loaded back into the plate group before microplate 2 is unloaded, the software will attempt to unload the first in the sequence, microplate 1.

If keeping the original order of labware is important, do one of the following:

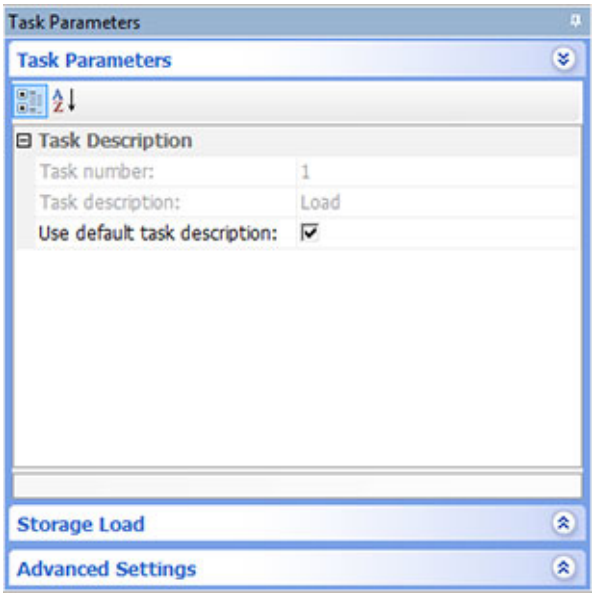
- Select the **Use original locations** option in the Load task parameters. The original storage locations will be reserved and the same labware order is preserved.

- When adding the Unload task preceding a Load task, select the **Remove plates from group when processed** option so that each labware is removed from the plate group (no longer a member of the plate group) as it is unloaded. This ensures that the labware is unloaded in the expected order. For example, a plate group consists of microplates 1 through 5. When microplate 1 is unloaded, it is removed from the plate group, so that microplate 2 becomes the first microplate in the group. If microplate 1 is loaded back into the plate group location before microplate 2 is unloaded, the software will attempt to unload microplate 2 instead of looking for microplate 1.

  When adding the Load task, add the labware to the original plate group. If you select the same plate group for the Load task, the labware will be loaded back into the plate group in the correct sequence.

Task parameters

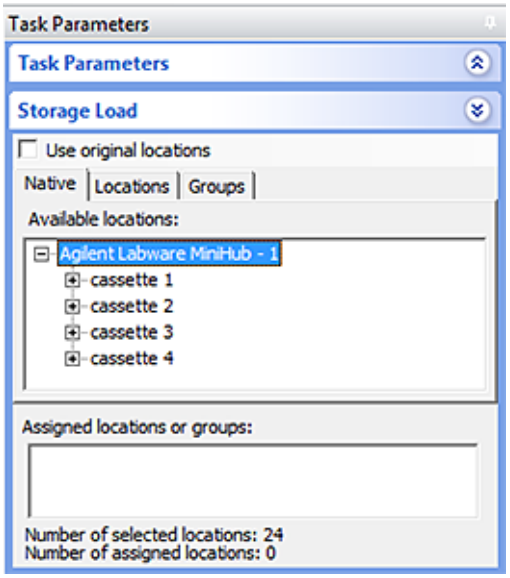
After adding the Load task at the desired point in the protocol, set the following **Task Parameters**:



Parameter	Description
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Storage Load parameters

After setting the Task Parameters, click **Storage Load**, and set the following parameters:



Parameter	Description
Use original locations	The option to move a set of labware back to its original storage location and maintain the original order in storage. For example, if a set of microplates were unloaded from Labware MiniHub A cassette 3, selecting this option in the Load task would move the microplates back to Labware MiniHub A cassette 3.
Native tab	The tab that allows you to select the storage locations from a list of known locations in the device file. Double-click a location in the **Available locations** area. The selected location appears in the **Assigned locations** area.
Locations tab	The tab that allows you to select locations from a list of location groups. The location groups are created in the Inventory Editor. Double-click a location group in the **Available locations** area. The selected location group appears in the **Assigned locations** area. To view or revise existing location groups, click **Edit location groups.**

Parameter	Description
Groups tab	The tab that allows you to select a plate group into which you want to add the labware. In addition to specifying the storage location during a Load task, you can add the labware to an existing plate group. Plate groups are created in the Inventory Editor. Double-click a plate group in the **Available groups** area. The selected plate group appears in the **Assigned locations** area. To view or revise existing plate groups, click **Edit plate groups.**

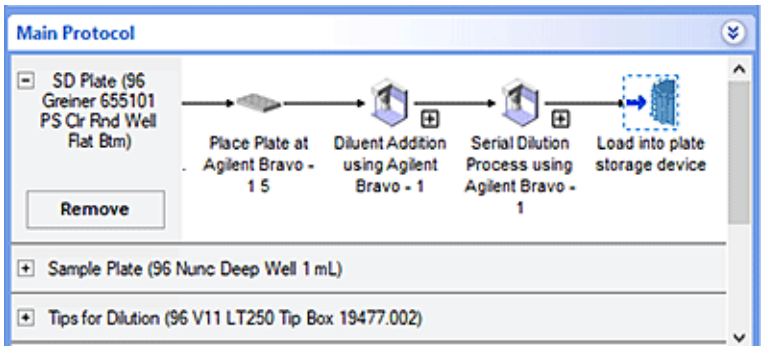
Example: Load microplates into the Labware MiniHub after processing

Goal

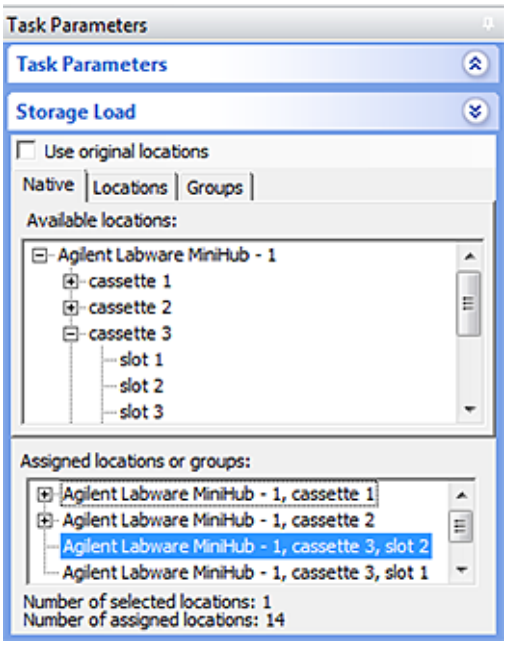
Load the destination microplates into Labware MiniHub cassettes 1 and 2, and cassette 3 slots 1 and 2 after the specified liquid-handling tasks.

Implementation

Add the Load task after the liquid-handling tasks as shown.



In the **Storage Load** area, click the **Native** tab. Select Labware MiniHub cassettes 1 and 2, and cassette 3 slots 1 and 2.



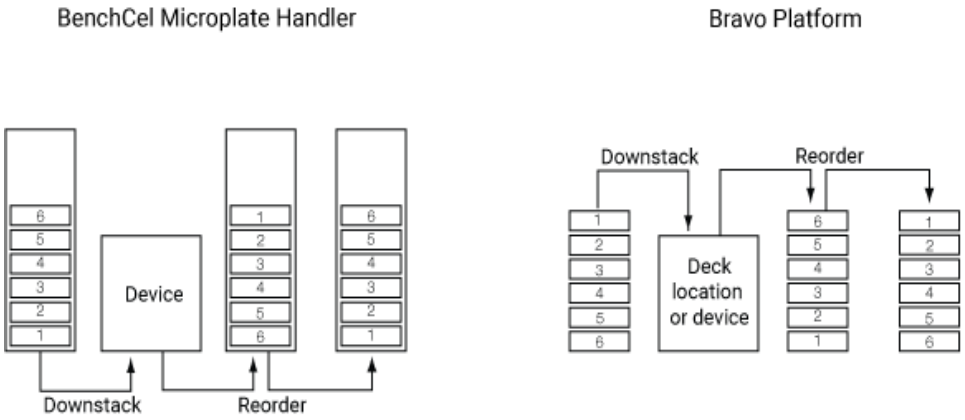
Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Setting up plate locations and plate groups in inventory manager	*VWorks Automation Control Setup Guide*
Unload task	"Unload" on page 438
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Reorder

Description

The Reorder task (Reorder) collects and incubates labware in a stack and then reverses the order of the labware in a second stack as shown in the following figure.



You use the Reorder task to:

- Ensure liquid transfers from different source microplates occur in the same order.
- Ensure equal incubation time across all the labware in a stack.
- Minimize evaporation in the labware.

In the example shown, labware is downstacked to a device or deck location where reagents are added. The labware is then upstacked to and incubated in the first of the two reorder stacks. The labware is then moved to a second stack so that the original stack order is maintained. If the labware moves on to a downstream task, the incubation time of the first labware is the same as the last labware to leave the stack. Without the Reorder task, the first labware to move to the next task has the shortest incubation time and the last labware has the longest incubation time.

Task is available for...	Task is available in...
BenchCel Microplate Handler	Main Protocol
Bravo Platform	Main Protocol, Bravo Subprocess

Requirements

General

At least two stacking locations are required for the Reorder task. For example, you need at least two BenchCel stacker heads or two Bravo deck locations.

Bravo Platform

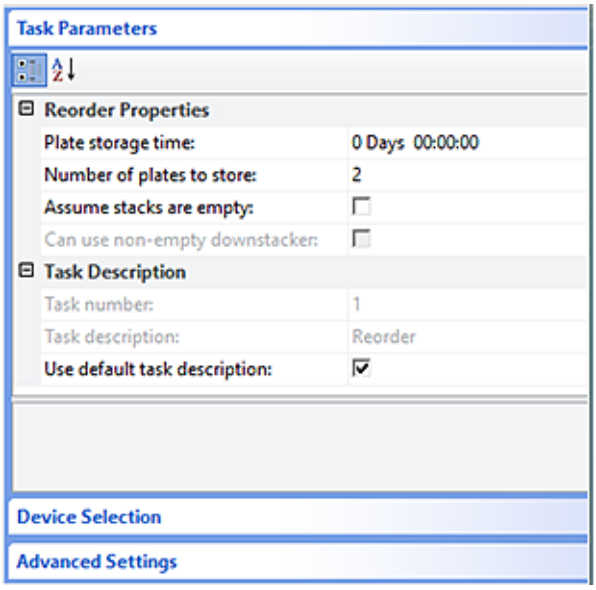
To use the Reorder task in a Bravo Subprocess, you must:

- Select the desired stacking options in the Bravo profile.
- Specify the maximum stack height in the Bravo deck location properties area.

- Add two Scan Stack tasks for each pair of stacking locations used in the task.

Task parameters

After adding the Reorder task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Plate storage time	The length of time to incubate the labware in the collection stack.
Number of plates to store	The maximum number of labware allowed in the collection stack for incubation. *Note:* This value can affect the timing of the incubation. For example, if the time taken to move all labware to the first reorder stack is greater than the time specified for the incubation, the first labware cannot be moved to the next task in time. To resolve this problem, reduce the number of labware to store and add more stackers.

Parameter	Description
Assume stacks are empty	The option to specify that the stack does not contain any labware. • If not selected (default), the Reorder task uses a stacker only if the stacker is empty or if the given device has no mechanism for determining if a stacker is empty or not. • If selected, the Reorder task assumes that any stacking device specified in the Device Selection area is empty, unless the stacker is used in a Downstack task. If the stacker is used in a Downstack task, the Reorder task assumes the stacker is empty only if it really is empty. The software automatically selects which stacker to use accordingly.
Can use non-empty downstacker	Available if Assume stacks are empty check box is selected. Select the check box for this parameter if you want a stacker, which is used by a Downstack task earlier in the same process lane, to also be available for use by the Reorder task. This option is available provided all process instances (plates) to be downstacked via the Downstack task have entered the system, even if the given stacker still has plates in it. *Note:* A compile error will occur if this parameter is selected, and one of the stackers used by the task is also used by a Downstack task in a different process lane.
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Device selection

You must select a device for Reorder tasks. If you have multiple devices of the same type, you can prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.

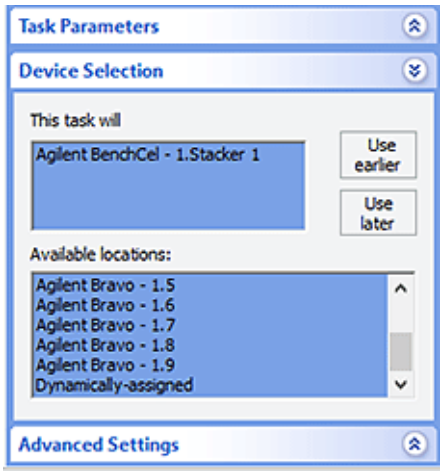
IMPORTANT The multiple devices must have the same setup and configuration.

After adding the Reorder task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1. Double-click a device or deck location in the **Devices available to perform task** area to add it to the **Devices involved in task** area.

   Note: If you want the software to dynamically move labware to available stackers, double-click **Dynamically assigned** in the **Devices available to perform task** area. You can check the run log to determine the location of the labware in the various stackers.
2. To prioritize its use, select the device in the **Devices involved in task** area, and then click **User earlier** or **Use later.** The devices that are higher in the list are favored by the software scheduler during the protocol run.



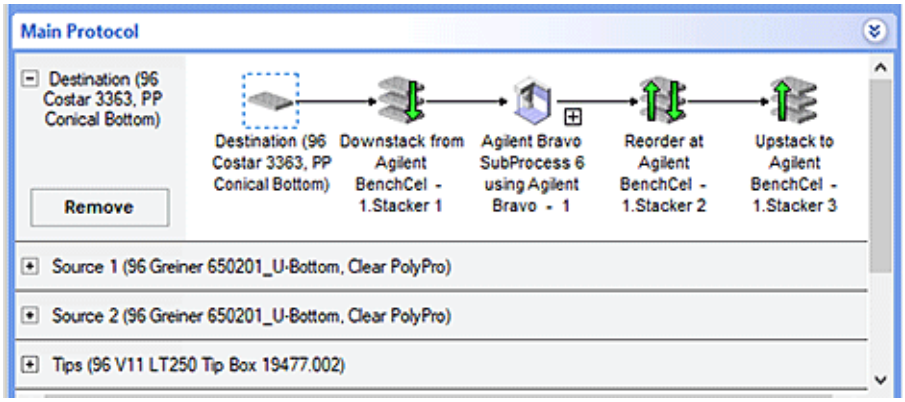
Example: Reorder microplates before reading them

Goal

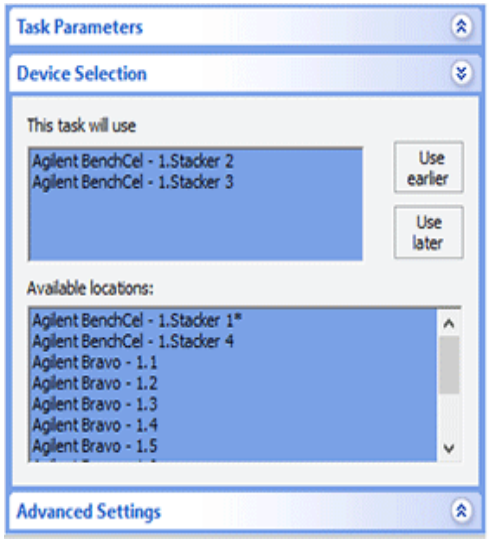
Aspirate contents from two source microplates, dispense into a destination microplate. Incubate the destination microplate for at least 20 minutes. Rearrange the microplate order to the original sequence, and then read the microplates.

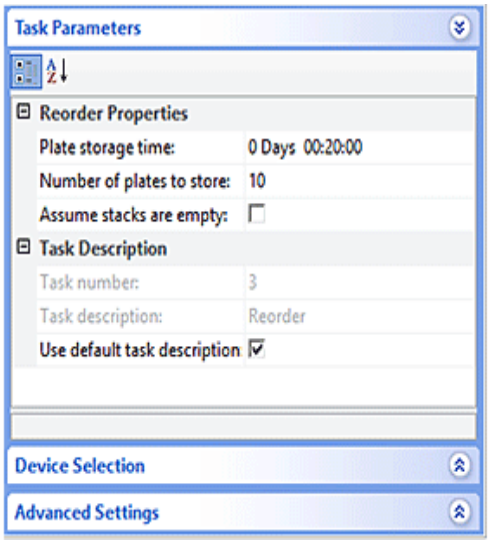
Implementation

In the protocol, add a liquid-handling subprocess to specify the tasks for adding the contents from the two source microplates into the destination microplate. Add a Reorder task after the liquid-handling tasks and before the Upstack task as shown in the following example.



For the Reorder task, select two stacking devices to use, and then set the Plate storage time to 20 minutes in the Task Parameters area as shown in the following examples.





Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Scan Stack task	"Scan Stack" on page 432
Downstack task	"Downstack" on page 414
Upstack task	"Upstack" on page 443
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Scan Stack

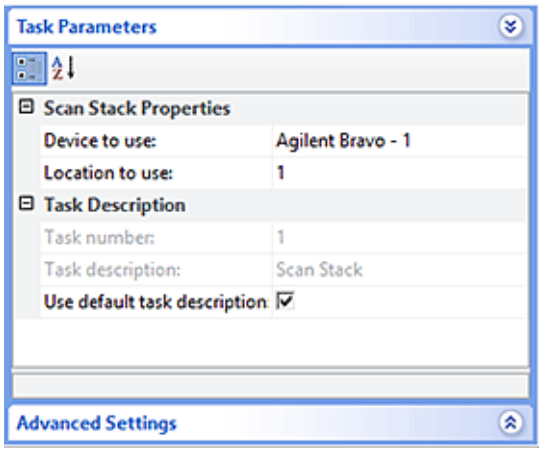
Description

The Scan Stack task (Scanstack) scans a specified Bravo deck location to confirm the presence or absence of labware and determine the height of the stack. You must add one Scan Stack task for each Downstack or Upstack task, and at least two Scan Stack tasks for each Reorder task in the Bravo Subprocess.

Task is available for...	Task is available in...
Bravo Platform	Startup Protocol Cleanup Protocol

Task parameters

After adding the Scan Stack task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Device to use	The Bravo Platform on which the stacking will occur.

Parameter	Description
Location to use	The the deck location to be scanned. If the deck location will be used for a Downstack task, the robot will scan the stack to confirm the presence of the stack and determine its height. If the deck location will be used for an Upstack task, the robot will scan the location to confirm that it is empty and ready to receive labware.
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

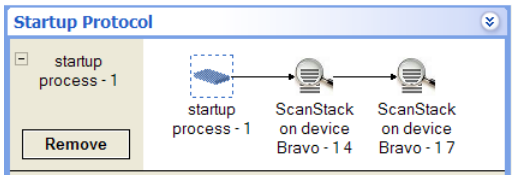
Example

Goal

Add the correct number of Scan Stack tasks to permit downstacking of microplates from deck location 4 for liquid-handling tasks. Add the correct number of Scan Stack tasks to permit uploading of those microplates to deck location 7 after processing.

Implementation

In the Startup Protocol, add two Scan Stack tasks: one to scan deck location 4 (where the microplates will be downstacked), and one to scan deck location 9 where the microplates will be upstacked.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Downstack task	"Downstack" on page 414
Upstack task	"Upstack" on page 443

For information about...	See...
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Storage Incubate

Description

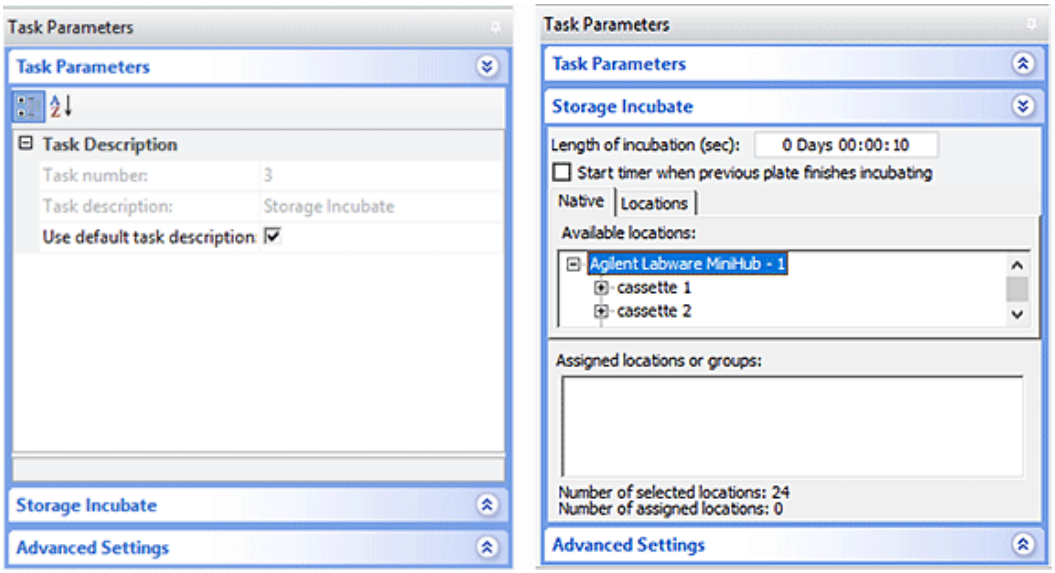
The Storage Incubate task (Storage Incubate) moves a defined set of labware into a storage device, leaves it there for a specified time period, and then moves it out of the storage device.

Note: To incubate labware on the Bravo Platform, use the Reserve Location task.

Task is available for...	Task is available in...
Labware MiniHub	Main Protocol

Task parameters

After adding the Storage Incubate task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Task Description	
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.
Storage Incubate	
Length of incubation	The length of time to incubate the labware.
Start timer when previous plate finishes incubation	The option to start timing the incubation after the previous labware has finished incubating.
Native tab	The tab that allows you to select the storage locations from a list of all the known locations in the device file. Double-click a location in the **Available locations** area. The selected location appears in the **Assigned locations** area.
Locations tab	The tab that allows you to select locations from a list of location groups. The location groups are created in the Inventory Editor. Double-click a location group in the **Available locations** area. The selected location group appears in the **Assigned locations** area. To view or revise existing location groups, click **Edit location groups.**

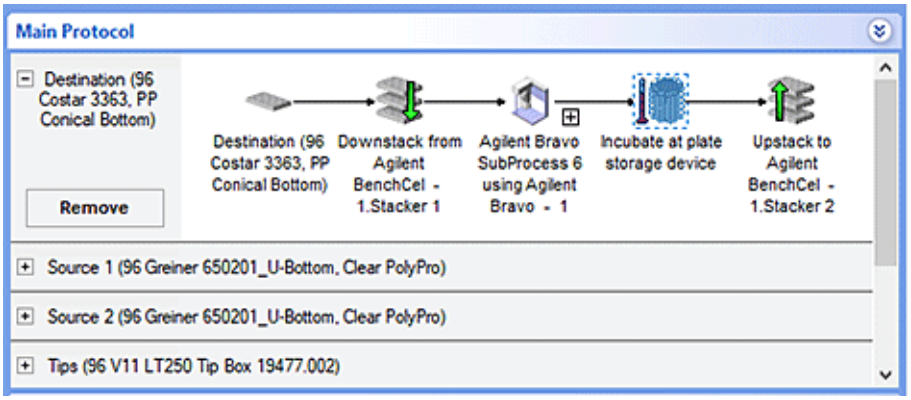
Example: Incubate destination microplates in Labware MiniHub

Goal

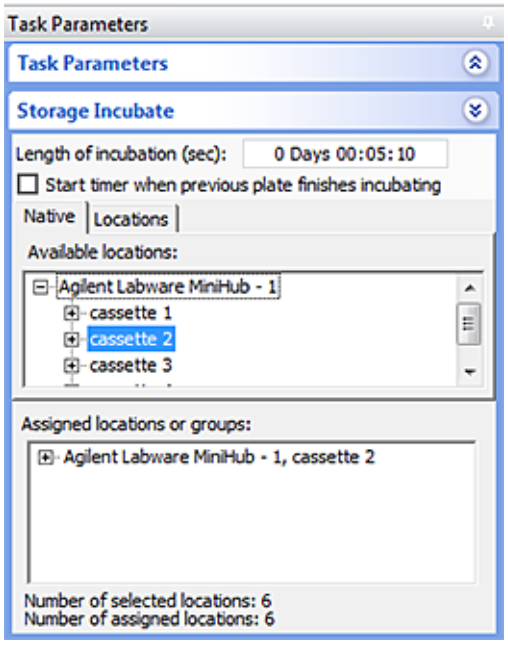
Aspirate contents from two source microplates and dispense into a destination microplate. Incubate the destination microplate in the Labware MiniHub for 5 minutes and 10 seconds before reading the microplates.

Implementation

In the protocol, add a liquid-handling subprocess to specify the tasks for moving contents from the two microplates into the destination microplates. Add a Storage Incubate task after the liquid-handling subprocess and before the Upstack task as shown in the following example.



In the Storage Incubate Task Parameters area, set the Length of incubation at 5 minutes 10 seconds as shown in the following example. The example assumes that cassette 2 is available for use.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Incubate task	"Incubate" on page 370
Reserve Location task	"Reserve Location" on page 401
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343

For information about...	See...
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Unload

Description

The Unload task () instructs a robot to move a defined set of labware out of a storage device.

Note: If you want to move a defined set of labware from one storage device to another, use the Unload task and Load task in sequence. For information about the Load task, see "Load" on page 422.

Task is available for...	Task is available in...
Labware MiniHub	Main Protocol

Storage location selections

When you add the Unload task, you can select the storage locations in one of the following ways:

- *Native locations*. Native locations are known locations in a device file. For example, if the Labware MiniHub is in the device file, you can select a cassette in that carousel for the Load task. When you select a cassette, all 16 slots in the cassette are available for the task. If you only have 10 microplates to load into that cassette, six of the slots will remain empty.
- *Location groups*. Location groups are device locations that are grouped together. For example, a location group might consist of 12 Labware MiniHub slots as follows: cassette 1 (5 slots), cassette 2 (5 slots), and cassette 3 (2 slots). If you want to load exactly 12 microplates, you can select that particular location group.
- *Plate groups*. Plate groups are labware that are grouped together. Each group member has location information. So when you add the Unload task and select a plate group, the software knows where to go to unload the labware as long as the information is in the inventory database.

Unload parameters that affect labware order

The software always unloads the first labware in the plate group. For example, if a plate group contains microplates 1 through 5, microplate 1 is unloaded, then microplate 2, and so on. If microplate 1 is loaded back into the plate group location before the labware 2 is unloaded, the software will attempt to unload the first in the sequence, labware 1.

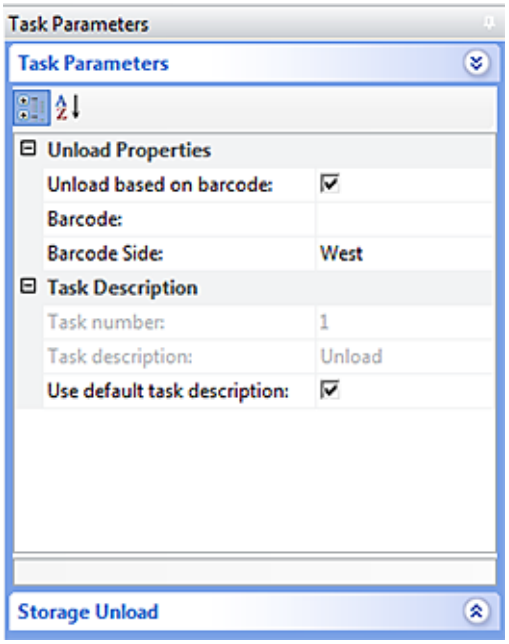
If you plan to load any unloaded plate groups back in storage devices in the same order, do one of the following:

- Select the **Use original locations** option in the Load task parameters. The original storage locations will be reserved and the same labware order is preserved.

- When adding the Unload task, select the **Remove plates from group when processed** option. When adding the Load task, either add the labware to the original plate group, or create a new plate group and add the labware to the new group. During the unload, the labware is removed from the plate group so that the software will seek the next labware in the group. If you select the same plate group for the Load task, the labware will be loaded back into the plate group at the bottom of the sequence.

Task parameters

After adding the Unload task at the desired point in the protocol, set the following **Task Parameters**:

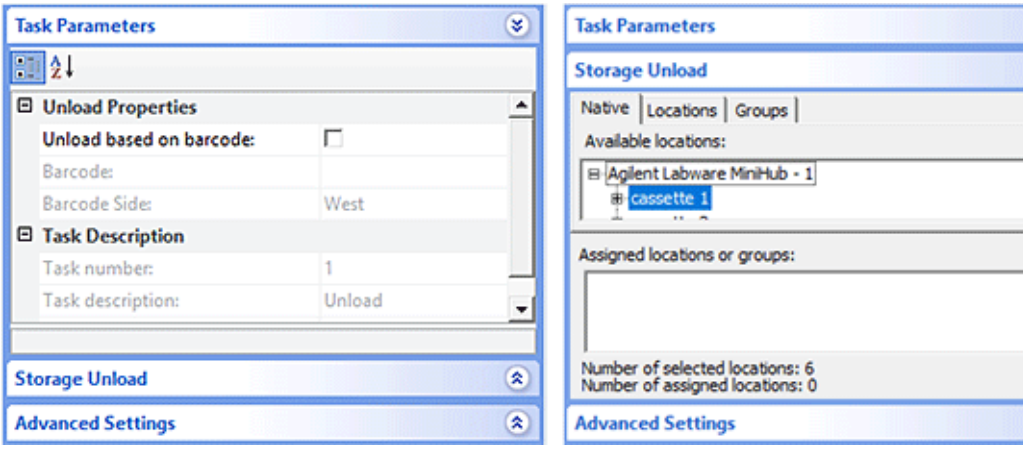


Parameter	Description
Unload based on barcode	The option to use the specified barcodes to search for and unload specific labware from storage. If you select this option, you must supply the barcodes in the **Barcode** box.
Barcode	The barcode of the labware you want to unload. You can supply one or more barcodes as follows: • Type the single barcode in the box. • Manually scan the single barcode label. • Type a JavaScript variable to supply a different barcode per labware instance. This box is only active if you selected **Unload base on barcode**.

Parameter	Description
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Storage Unload parameters

After setting the Task Parameters, click **Storage Unload**, and set the following parameters:



Parameter	Description
Native tab	The tab that allows you to select the storage locations from a list of known locations in the device file. Double-click a location in the **Available locations** area. The selected location appears in the **Assigned locations** area.
Locations tab	The tab that allows you to select locations from a list of location groups. The location groups are created in the Inventory Editor. Double-click a location group in the **Available locations** area. The selected location group appears in the **Assigned locations** area. To view or revise existing location groups, click **Edit location groups.**

Parameter	Description
Groups tab	The tab that allows you to select a plate group that you want to move out of storage. Double-click a plate group in the **Available groups** area. The selected plate group appears in the **Assigned locations** area. To view or revise existing plate groups, click **Edit plate groups.**
Remove plates from group when processed	The option to remove the specified labware from a plate group after the labware is moved out of storage.

Note: A compiler error will appear if you try to unload from a plate group (Groups tab) and another storage location type (Native tab or Locations tab).

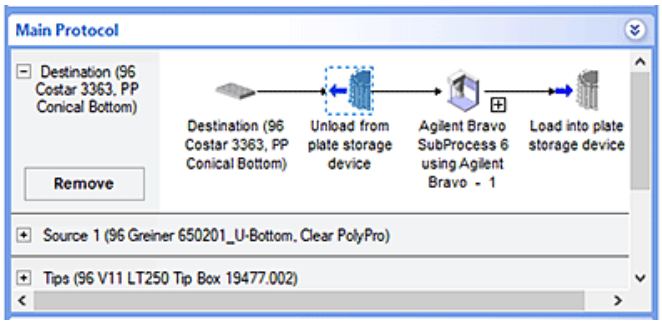
Example: Unload microplates from Labware MiniHub for processing

Goal

Unload the specified microplates from Labware MiniHub cassette 1 in preparation for liquid-handling tasks. In this example, barcodes will not be used to unload the microplate.

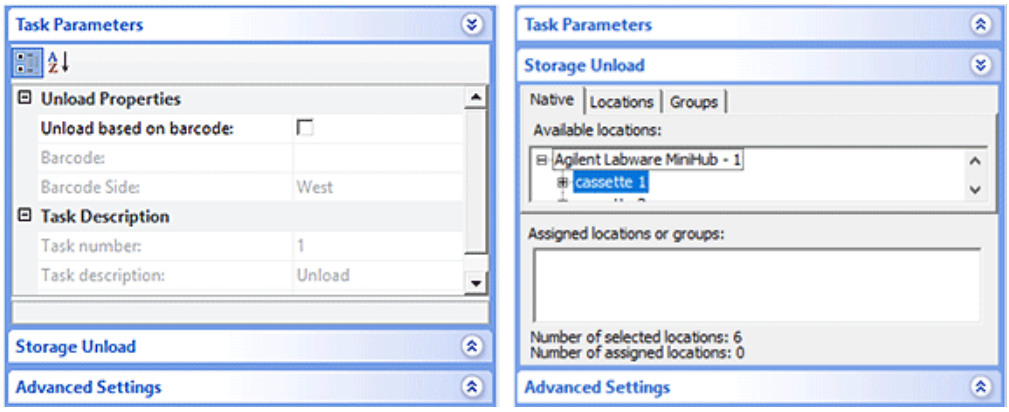
Implementation

In the protocol, add the Unload task before the liquid-handling tasks as shown.



In the Task Parameters area, clear the **Unload based on barcode** check box.

In the Storage Unload area, click the Native tab. Select cassette 1 in the Labware MiniHub to use the cassette.



Related information

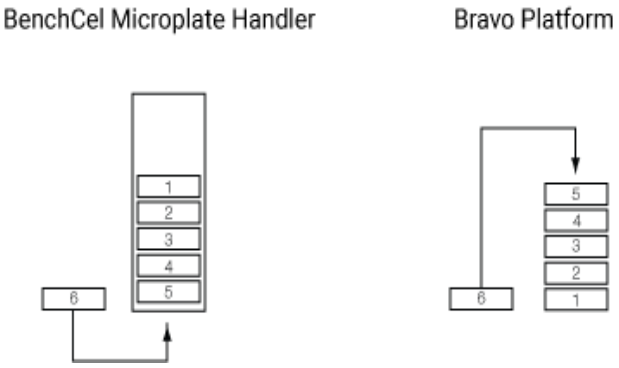
For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Setting up plate locations and plate groups in inventory manager	*VWorks Automation Control Setup Guide*
Load task	"Load" on page 422
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Upstack

Description

The Upstack task (Upstack) moves labware as follows:

- Into the bottom of stacking devices such as the BenchCel Microplate Handler
- Onto the top of a stack of labware at a Bravo deck location



For example, you can use the Upstack task to move a microplate from a platepad into the BenchCel Microplate Handler.

You can use a single Upstack task to move labware into multiple stacking devices. For example, if three stackers are available, the upstack task can dynamically move microplates into any of the three stackers.

Task is available for...	Task is available in...
BenchCel Microplate Handler	Main Protocol
Bravo Platform	Main Protocol, Bravo Subprocess

Requirements

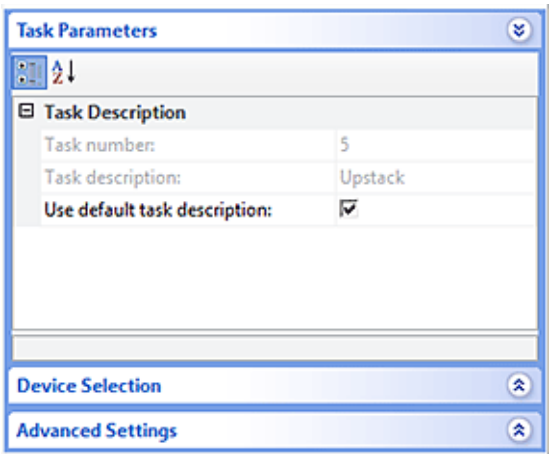
Bravo Platform

To use the Upstack task in a Bravo Subprocess, you must:

- Select the desired stacking options in the Bravo profile.
- Specify the maximum stack height in the Bravo deck location properties area.
- Add a Scan Stack task for each Upstack task.

Task parameters

After adding the Upstack task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Device selection

You must select a device for Upstack tasks. If you have multiple devices of the same type, you can prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.

IMPORTANT

The multiple devices must have the same setup and configuration.

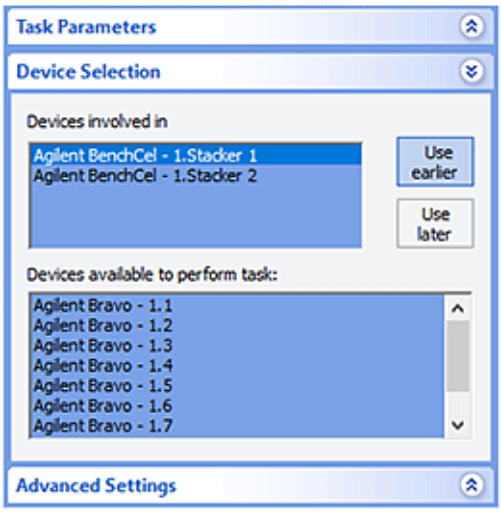
After adding the Upstack task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1 Double-click a device or deck location in the **Devices available to perform task** area to add it to the **Devices involved in task** area.

Note: If you want the software to dynamically move labware to available stackers, double-click **Dynamically assigned** in the **Devices available to perform task** area. You can check the run log to determine the location of the labware in the various stackers.

2 To prioritize its use, select the device in the **Devices involved in task** area, and then click **User earlier** or **Use later.** The devices that are higher in the list are favored by the software scheduler during the protocol run.



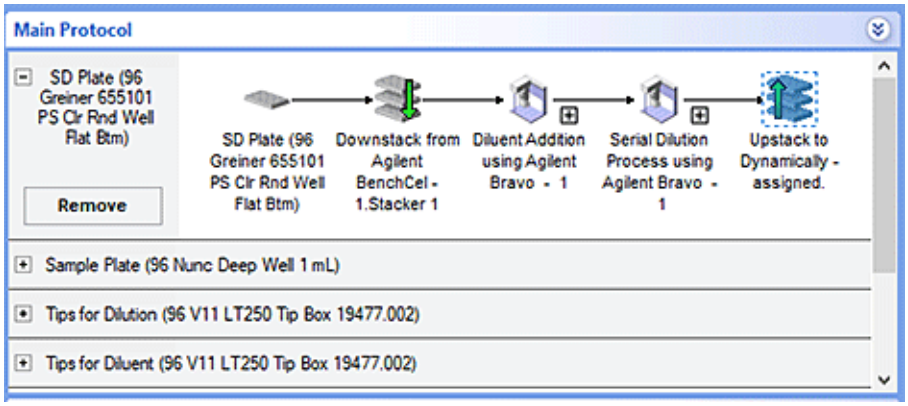
Example 1: Dynamically upstack microplates into a stacking device

Goal

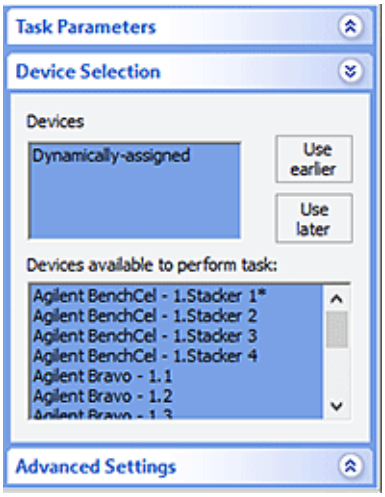
After the Destination Plate is finished with the Bravo liquid-handling process, upstack it to any available stacker.

Implementation

Add a Upstack task after the Bravo Subprocess task that contains the liquid-handling sub-routine.



The device selection is Dynamically assigned:



Example 2: Stacking on the Bravo Platform

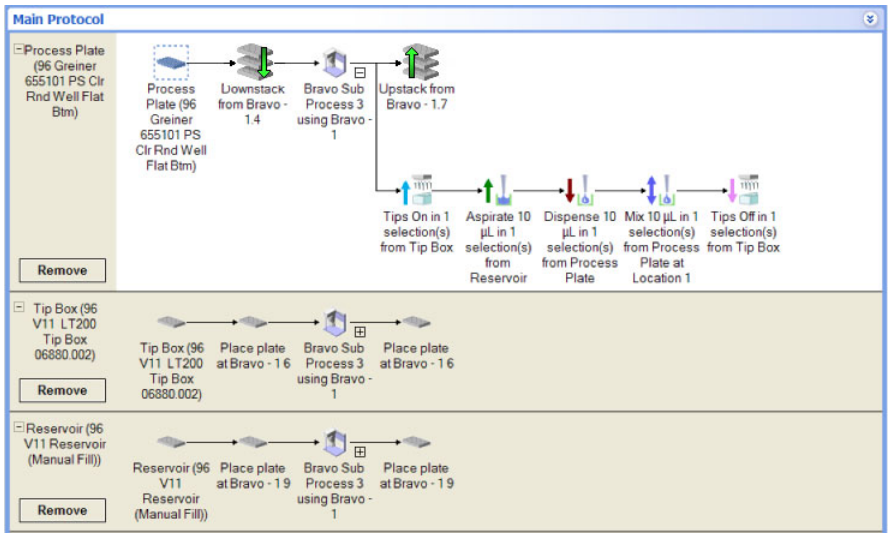
Goal

Downstack microplates for liquid-handling process, then upstack them.

Implementation

The protocol is set up as follows:

- A stack of microplates (Process Plates) is at deck location 4.
- A tip box (Tip Box) is configured at deck location 6.
- A Manual Fill Reservoir (Reservoir) is configured at deck location 9.
- The Startup Protocol (not shown) contains Scan Stack tasks for deck locations 4 (where the stack starts) and 7 (where the stack will end up after upstacking).



The protocol performs the following:

1 The robot scans deck location 4 to confirm the presence of the stack and determines the number of microplates. (The Scan Stack task in the Startup Protocol, not shown, performs this task.)

2 The robot scans deck location 7 to confirm the absence of labware. (The Scan Stack task in the Startup Protocol, not shown, performs this task.)

3 The top-most Process Plate is downstacked from deck location 4 to deck location 1.

   By default, when the <auto-select> location is selected for a process plate task, the process plates are always placed or downstacked at deck location 1. If deck location 1 is occupied, the process plate will be placed or downstacked at the next-available location.

4 Tips are installed on the pipette head at deck location 6.

5 The robot aspirates solution from the Reservoir at deck location 9.

6 The robot dispenses the solution into the Process Plate at deck location 1.

7 The robot mixes the solution in the Process Plate at deck location 1.

8 The pipette tips are removed at the Tip Box at deck location 6.

9 The Process Plate at deck location 1 is upstacked to deck location 7.

10 Steps 1 through 8 is repeated for each Process Plate stacked at deck location 4.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Scan Stack task	"Scan Stack" on page 432
Downstack task	"Downstack" on page 414
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

9 Setting parameters for liquid-handling tasks

This chapter describes the Bravo liquid-handling tasks you can add in a protocol:

- "SubProcess (Agilent Bravo)" on page 450
- "Aspirate (Agilent Bravo)" on page 455
- "AM Aspirate (Agilent Bravo)" on page 461
- "AM Dispense (Agilent Bravo)" on page 471
- "AM Mix (Agilent Bravo)" on page 481
- "AM Cartridges Off (Agilent Bravo)" on page 489
- "AM Cartridges On (Agilent Bravo)" on page 491
- "AM Wash Syringes (Agilent Bravo)" on page 494
- "Assemble Vacuum (Agilent Bravo)" on page 499
- "Dilute to Final Volume (Agilent Bravo)" on page 504
- "Disassemble Vacuum (Agilent Bravo)" on page 512
- "Dispense (Agilent Bravo)" on page 514
- "Dispense to Waste (Agilent Bravo)" on page 521
- "Hit Pick Replication (Agilent Bravo)" on page 526
- "Initialize Axis (Agilent Bravo)" on page 558
- "Mix (Agilent Bravo)" on page 559
- "Move and Filter Plate (Agilent Bravo)" on page 568
- "Pin Tool (Agilent Bravo)" on page 574
- "Pipette Under Collar Begin and End (Agilent Bravo)" on page 584
- "Pump Reagent (Agilent Bravo)" on page 591
- "Serial Dilution (Agilent Bravo)" on page 595
- "Set Head Mode (Agilent Bravo)" on page 601
- "Set Temperature (Agilent Bravo)" on page 609
- "Set Temperature - Position (Agilent Bravo)" on page 611
- "Shake (Agilent Bravo)" on page 613
- "Shake Plate (Agilent Bravo)" on page 616
- "Tips Off (Agilent Bravo)" on page 619
- "Tips On (Agilent Bravo)" on page 623
- "Toggle Vacuum (Agilent Bravo)" on page 626
- "Wash Tips (Agilent Bravo)" on page 632

SubProcess (Agilent Bravo)

Description

The Subprocess (Agilent Bravo) ( SubProcess (Agilent Bravo)) task indicates the start of a protocol subroutine that employs the Bravo Platform. Within the subprocess, you can add tasks that are unique to the device. You can expand or collapse the subprocess to show or hide the subprocess tasks.

IMPORTANT

All Bravo liquid-handling tasks must be added within a Bravo Subprocess.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol

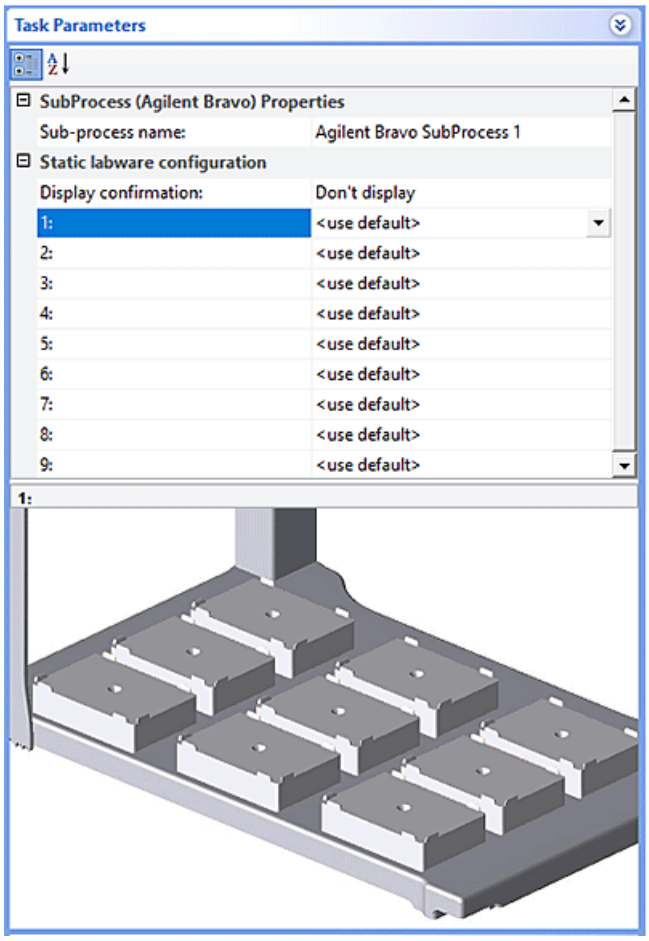
Task parameters

IMPORTANT

Agilent recommends that you use the concepts and procedure in "Configuring labware" on page 53 when writing new protocols.

After adding the Subprocess task at the desired point in the protocol, you have the option of assigning static labware to locations on a device. You assign static labware in the subprocess to override the static labware configuration in the Startup Protocol. For more information about static labware, see "Planning labware use" on page 24.

To assign static labware to locations on the device, set the following parameters in the **Task Parameters** area:



Parameter	Description
Subprocess name	The name of the subprocess. Select from the list of available subprocesses currently in the protocol.
Display confirmation	The option to display a message at the beginning of the protocol run and the subprocess to remind you to verify the physical locations of the labware match what you specified in the software.
Location n	The Bravo deck locations. Select the labware for the specific locations.

Device selection

You must select a device for Subprocess tasks. If you have multiple devices of the same type, you can:

- Prioritize the list of devices for the task. If the first device in the list is busy, the software will automatically use the next device in the list. If all of the devices in the list are busy, the task that needs the device will wait until one becomes available.

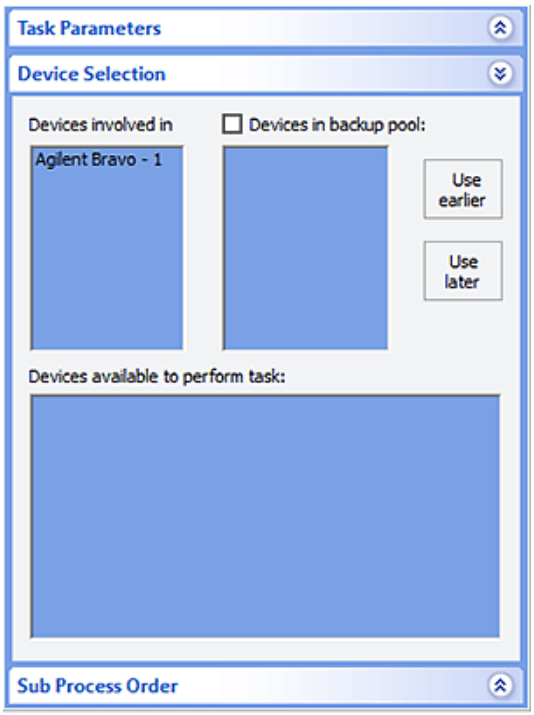
- Set up a backup pool. If the primary device encounters an error, the software will automatically use the next device in the list. However, if all of the devices in the list are in an error state, the software will automatically use the device in the backup pool.

IMPORTANT The multiple devices must be configured with the same static labware and accessories, and all the accessories must have identical setups.

After adding the Subprocess task at the desired point in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

1 Double-click the desired device (deck location) in the **Devices available to perform task** area to move it to the **Devices involved in task** area. If you have multiple devices of the same type, you can move them to the **Devices involved in task** area.



2 If you have multiple devices in the **Devices involved in task** area, select a device, and then click **Use earlier** or **Use later** to prioritize it.

3 *Optional.* Select backup devices to use in case all of the devices in the **Devices involved in task** area encounter an error.

   a Select **Devices in backup pool.**

   b Drag one or more devices from the **Devices available to perform task** area to the **Devices in backup pool** area.

   c If you have multiple devices in the backup pool, select a device in the **Devices in backup pool** area, and then click **Use earlier** or **Use later** to prioritize it.

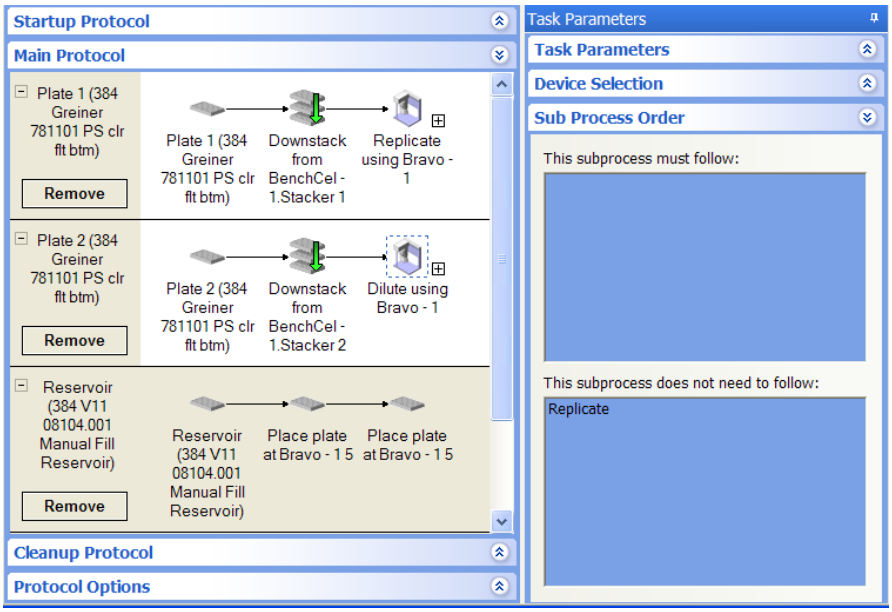
Subprocess Order

If more than one sub-process uses the same configured labware, and the sub-processes are in different protocol processes, you can specify the sequence in which the sub-processes will be performed.

To specify the sequence in which the sub-processes will be performed on the same configured labware:

1 In the protocol process, select the sub-process that contains the task that uses the configured labware.

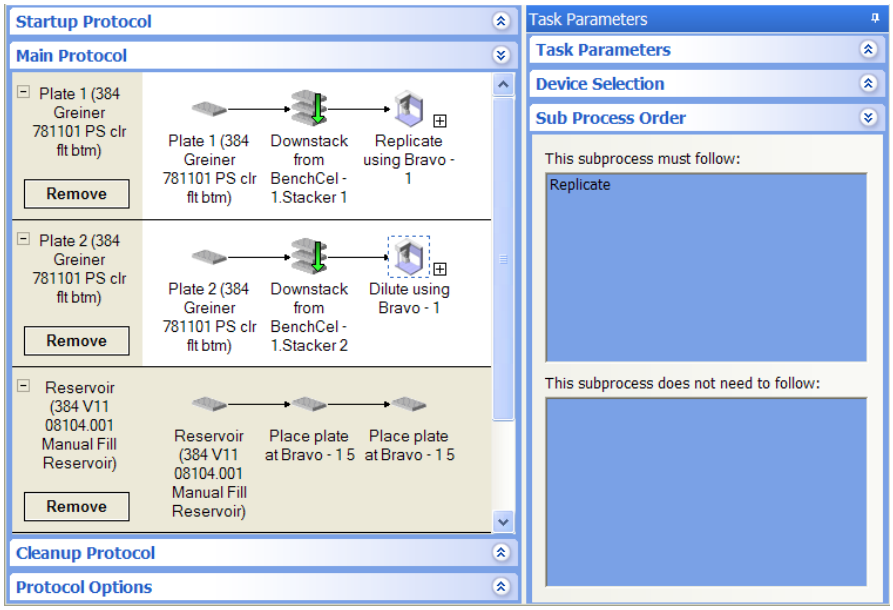
In the following example, the Bravo Subprocess named Dilute is selected.



2 In the **Task Parameters** area, click **Sub Process Order.**

3 In the **Sub Process Order** area, double-click the subprocess names to rearrange the order.

In the following example, the Dilute subprocess is selected. So the task sequence will be Replicate subprocess, followed by the Dilute subprocess.



Related information

For information about...	See...
Configured labware	"Configuring labware" on page 53
Static labware	"Planning labware use" on page 24
Startup Protocol	"Setting up Startup and Cleanup Protocol processes" on page 75
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Aspirate (Agilent Bravo)

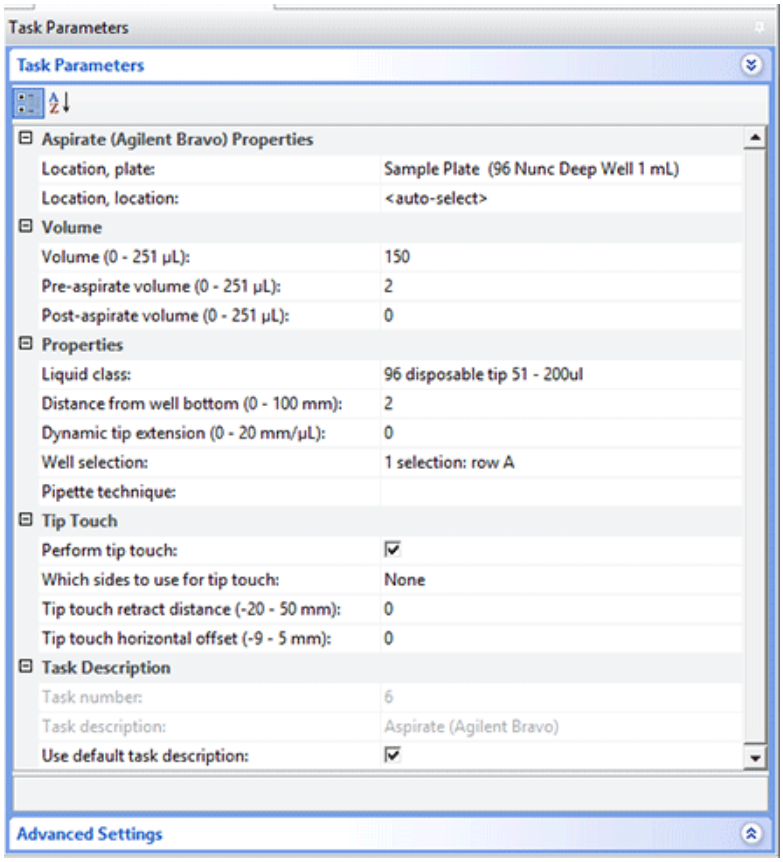
Description

The Aspirate (Agilent Bravo) (Aspirate (Agilent Bravo)) task draws liquid from a microplate or reservoir.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Task parameters

After adding the Aspirate task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Volume (0–251 µL)	The volume of liquid to be drawn into each pipette tip.
Pre-aspirate volume (0–251 µL)	The volume of air to be drawn before the pipette tips enter the liquid.

Parameter	Description
Post-aspirate volume (0–251 µL)	The volume of air to be drawn after the liquid is drawn.
Liquid class	The pipetting speed and accuracy. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Distance from well bottom (0–100 mm)	The distance between the end of the pipette tips and the well bottoms during the Aspirate task. If you specify dynamic tip extension, this is the distance at the end of the Aspirate task. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the tips at the correct distance from the well bottom.)
Dynamic tip extension (0–20 mm/µL)	The rate at which the pipette head moves during the Aspirate task. The software calculates the distance over which the tips will move without crashing. Use dynamic tip extension to prevent spills as the pipette tips displace the liquid. To move the tips: • *At the same rate as the volume change*. Calculate dynamic tip extension (DTE) as follows: DTE = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTE > 1/A • *Slower than the volume change*. DTE < 1/A The starting and ending positions can be calculated as follows: $(V_{aspirated} * DTE) + Distance_{well\ bottom}$ *Note:* Instead of a negative aspirated volume, the software automatically moves downward toward the well bottom with each aspirate action.
Pipette technique	The pipette location offset you want to use for the Aspirate task. The list of pipette techniques are defined in the Pipette Technique Editor.
Perform tip touch	The option to touch the pipette tip on one or more sides of the well.

Parameter	Description
Which sides to use for tip touch	The side or sides of the well to use during tip touch: North, South, East, West, North/South, West/East, West/East/South/North.
Tip touch retract distance (–20 to 50 mm)	The vertical distance for the pipette tips to rise before touching the sides of the wells.
Tip touch horizontal offset (–9 to 5 mm)	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. For example, if you set a value of: • 0, the tips move a horizontal distance equal to the well radius • > 0, the tips attempt to move past the well radius, which results in a more forceful tip touch • < 0, the tips move a distance less than the radius of the well, resulting in a lighter tip touch

Quadrant pattern well selection

Instead of a column- or row-wise pattern, you can select a quadrant pattern during well selection.

The quadrant pattern option is available only if:

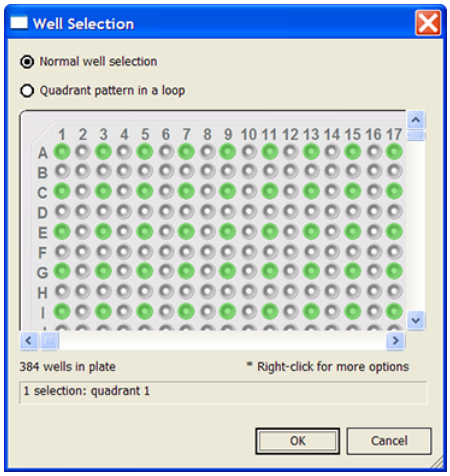
- The number of channels in the pipette head (or pins in a pin tool) is fewer than the number of wells in the microplate. For example, you can use a 96-channel pipette head to dispense liquid into a 384-well microplate or 1536-well microplate.
- All the channels are selected in the Set Head Mode task when using a pipette head. (The Set Head Mode task is not an option when using a pin tool).
- The liquid-handling task is inside a loop.

IMPORTANT

If you select a quadrant pattern, specifications in the Well Selection dialog box will override `task.Wellselection` values assigned in the Advanced Settings area.

To select a quadrant pattern:

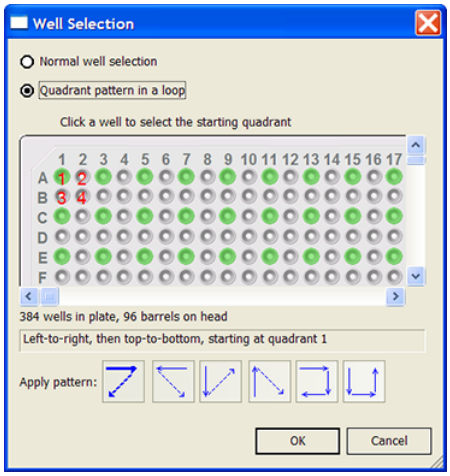
1 In the **Task Parameters** area, click the **Well selection** parameter box, and then click the Browse button. The Well Selection dialog box opens. By default, the **Normal well selection** option is selected. This option is used for column- and row-wise liquid-handling patterns.



2 Select **Quadrant pattern in a loop.** The contents of the dialog box change. Notice the following:

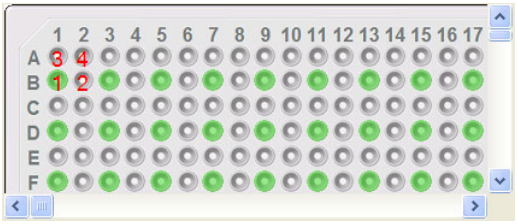
- Red numbers (1 through 4) appear on wells A1, A2, B1, and B2. The numbers indicate the pipetting sequence: 1 is the starting well, and 4 is the last well. In the following example, the sequence is A1, A2, B1, B2.
- Green wells indicate the starting well in the pipetting sequence.
- Pattern buttons at the bottom of the dialog box indicate the movement of the pipette channels. The movement description is provided in the text box above the buttons.

  Note: The last two patterns are unavailable if a group contains 16 wells. For example, the last two patterns are not available if you have a 96-well pipette head and a 1536-well microplate.



3 Select the starting well. The well becomes green and is labeled 1.

In the following example, the third quadrant (B1 well) is selected.



4 Click a pattern button to specify the pipette channel movement. After you click a pattern, the red numbers in the graphic are updated to show the sequence.

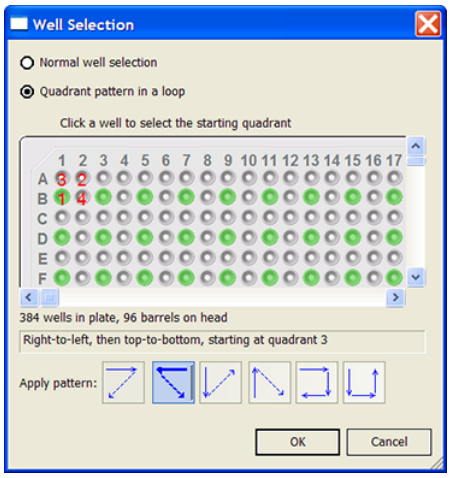
In the following example, the second pattern is selected (right-to-left, then top-to-bottom). The third quadrant (B1) is the starting well. The resulting movement is:

Quadrant 3 (B1)

Quadrant 2 (A2)

Quadrant 1 (A1)

Quadrant 4 (B2)



5 When you are finished, click **OK** to save the changes and return to the VWorks window.

Example: Aspirate from a source microplate on the Bravo Platform

Goal

Aspirate contents from a source microplate (Source 1) and dispense into a destination microplate.

Implementation

The Bravo deck is physically set up as follows:

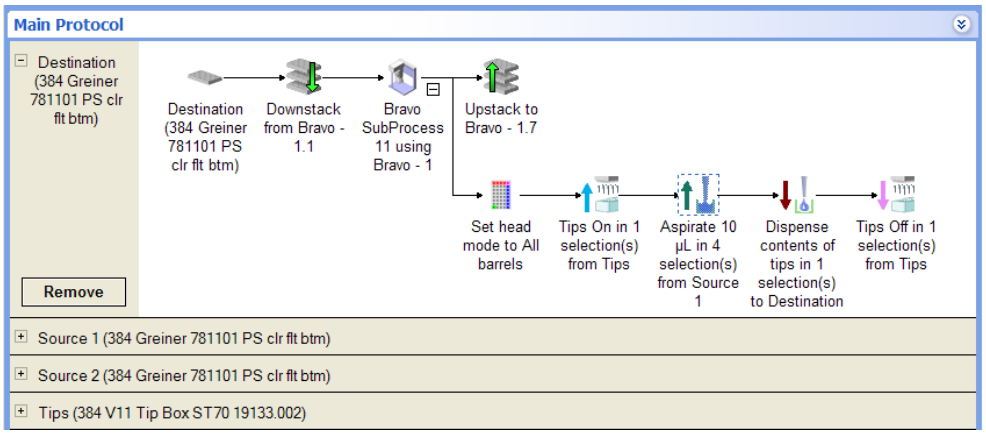
- The destination microplates are at Bravo deck location 1.
- The source microplate is at deck location 2.
- The tip box is at deck location 9.

In the protocol, the following are added:

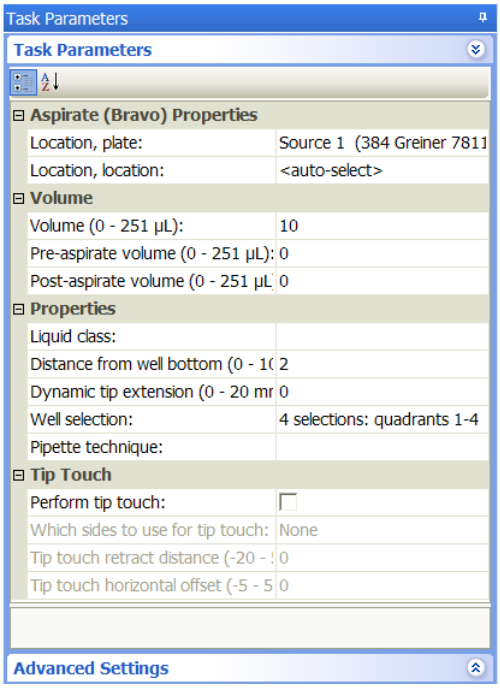
- Process for the destination microplate
- Configured labware for the source microplate

- Configured labware for the tip box

In the Destination plate process, a Bravo subprocess is added. Within the subprocess, an Aspirate task and a Dispense task are added as shown in the following example.



In the Aspirate Task Parameters area, Source 1 is selected, because the goal is to aspirate from that microplate.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67

For information about...	See...
Liquid classes	*VWorks Automation Control Setup Guide*
Pipette techniques	"Specifying pipetting techniques" on page 691
Dispense task	"Dispense (Agilent Bravo)" on page 514
Set Head Mode task	"Set Head Mode (Agilent Bravo)" on page 601
Tips On task	"Tips On (Agilent Bravo)" on page 623
Tips Off task	"Tips Off (Agilent Bravo)" on page 619
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

AM Aspirate (Agilent Bravo)

Description

The AM Aspirate (Agilent Bravo) (

) task is designed for aspirating fluids through AssayMAP Bravo cartridges that are mounted on the Bravo 96AM Head.

Task is available for...	Task is available in...
Bravo Platform with Bravo 96AM Head	Main Protocol, Bravo Subprocess

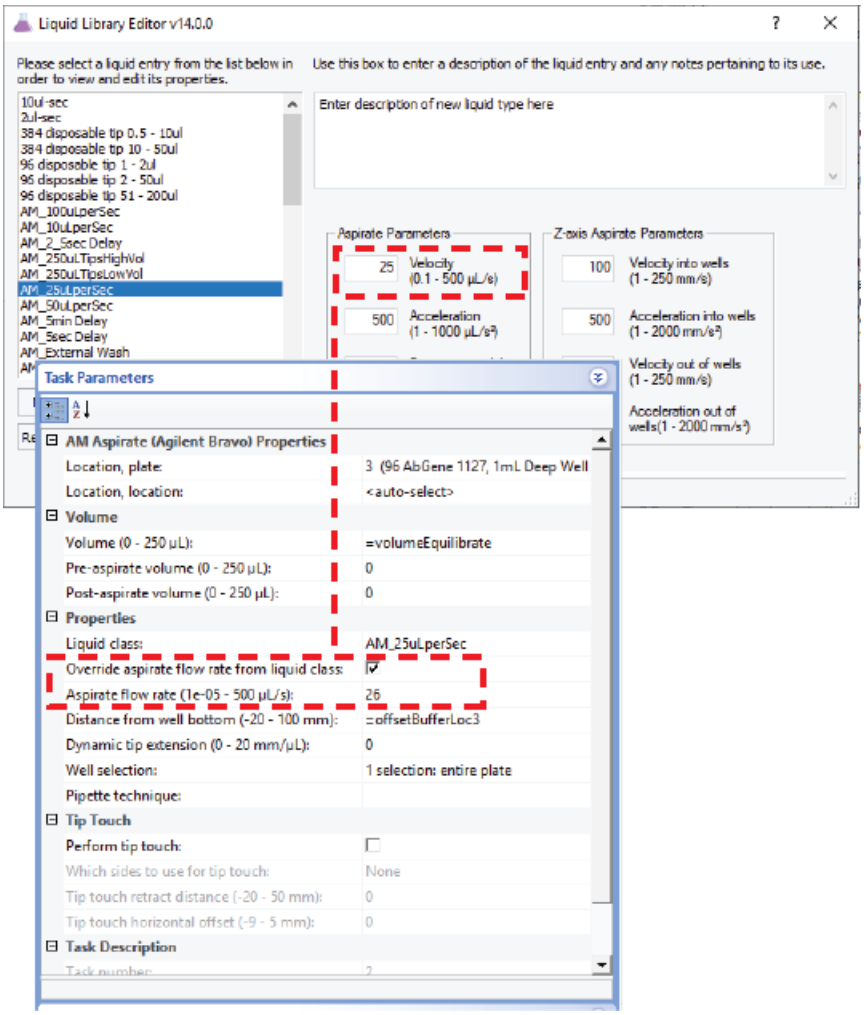
The AM Aspirate (Bravo) task has the same task parameters as the Aspirate (Bravo) task, except for the following parameters:

- Override aspirate flow rate from liquid class

  Precise flow-rate control is critical for the performance of the AssayMAP Bravo cartridges. The option to override the aspirate flow rate (velocity) in the liquid class enables an operator to change the flow rate from experiment to experiment for rapid method development without modifying the liquid class. Likewise, the flow rate can vary for different AM Aspirate tasks that use the same liquid class in a given protocol.
- Distance from well bottom (-20–100 mm)

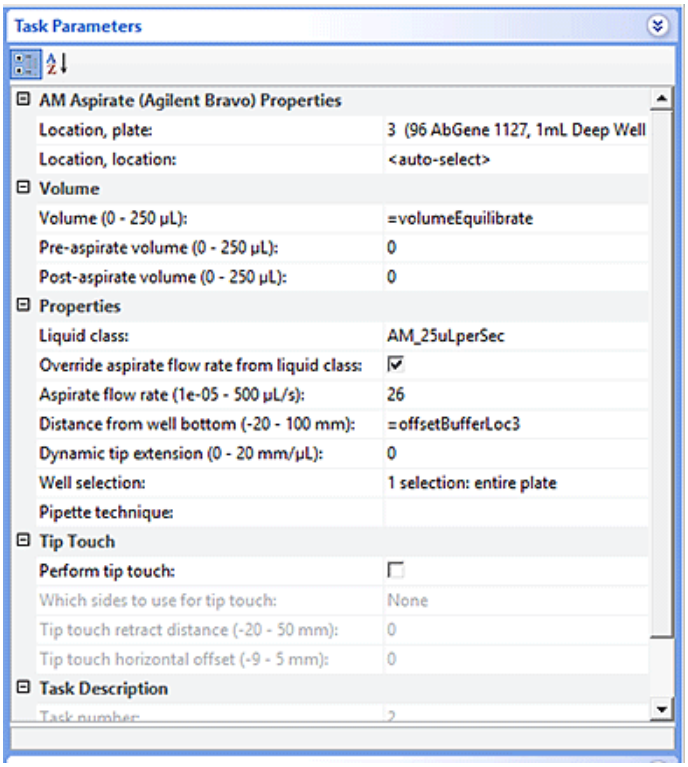
  The range includes negative values, which is critical for steps that require using the bare probes to aspirate from the top portion of unmounted cartridges.

Figure Liquid class velocity setting in the Liquid Library Editor and flow rate override option



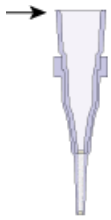
Task parameters

After adding the AM Aspirate task at the desired point in the protocol, set the following parameters in the **Task Parameters** area.



Parameter	Description
Volume (0–250 µL)	The volume of liquid to be drawn into each probe, cartridge, or tip.
Pre-aspirate volume (0–250 µL)	The volume of air to be drawn before the probes, cartridges, or tips enter the liquid.
Post-aspirate volume (0–250 µL)	The volume of air to be drawn after the liquid is drawn.
Liquid class	The pipetting speed and accuracy. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Override aspirate flow rate from liquid class	The option to override the aspirate velocity in the specified liquid class. Selecting this option enables you to specify a value for the aspirate flow rate without changing the liquid class.

Parameter	Description
Aspirate flow rate (1e-05 – 500 µL/s)	The numerical value or the JavaScript variable that will override the aspirate velocity setting in the liquid class. A JavaScript variable enables the value to be assigned later. For example, using a VWorks form, an operator could easily change the flow rate for an aspirate step in increments from as low as 1 µL/min up to 2000 µL/min or more using the same liquid class. If the task is included in a VWorks macro, a JavaScript variable enables you to change the value for the task at the macro level. **IMPORTANT** The software requires that the flow rate value be in microliters per second (µL/s) at run time. If you want an operator to enter the value in microliters per minute (µL/min), you can use scripting to convert the values for the software to use.

Parameter	Description
Distance from well bottom (-20–100 mm)	The distance between the tips of the probes, cartridges, or disposable tips and the *well bottoms* during the AM Aspirate task. If you specify dynamic tip extension, this is the distance at the end of the AM Aspirate task. Use a positive number for tasks that are performed with mounted cartridges, mounted tips, or bare probes at labware other than a cartridge rack. If the task is performed in the upper cup of unmounted cartridges, you can use a negative number for the parameter value. In this case, the *well bottom* is the top of the cartridge cup, as shown in the following figure.  For example, a negative number enables a cup wash step while the cartridges are in the cartridge rack before they are mounted on the Bravo 96AM Head. **IMPORTANT** You can use a negative number for this parameter only if the AM Aspirate task is performed in cartridges that are located in a cartridge rack. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the tips at the correct distance from the well bottom.

Parameter	Description
Dynamic tip extension (0–20 mm/µL)	The rate at which the Bravo 96AM Head moves during the AM Aspirate task. The software calculates the distance over which the tips will move without crashing. Use dynamic tip extension to prevent spills as the pipette tips displace the liquid. To move the tips: • *At the same rate as the volume change*. Calculate dynamic tip extension (DTE) as follows: DTE = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTE > 1/A • *Slower than the volume change*. DTE < 1/A The starting and ending positions can be calculated as follows: $(V_{aspirated} * DTE) + Distance_{well\ bottom}$ *Note:* Instead of a negative aspirated volume, the software automatically moves downward toward the well bottom with each aspirate action.
Pipette technique	The pipette location offset you want to use for the AM Aspirate task. The list of pipette techniques are defined in the Pipette Technique Editor.
Perform tip touch	The option to touch the pipette tip on one or more sides of the well.
Which sides to use for tip touch	The side or sides of the well to use during tip touch: North, South, East, West, North/South, West/East, West/East/South/North.
Tip touch retract distance (–20 to 50 mm)	The vertical distance for the pipette tips to rise before touching the sides of the wells.

Parameter	Description
Tip touch horizontal offset (–5 to 5 mm)	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. For example, if you set a value of: • 0, the tips move a horizontal distance equal to the well radius • > 0, the tips attempt to move past the well radius, which results in a more forceful tip touch • < 0, the tips move a distance less than the radius of the well, resulting in a lighter tip touch

Quadrant pattern well selection

A quadrant is an evenly spaced array of locations that are accessible by the tips on a pipette head.

Note: For the Bravo 96AM Head, the quadrant pattern option is available only if you are using 384-well microplates.

Instead of a column- or row-wise pattern, you can select a quadrant pattern during well selection.

The quadrant pattern option is available only if:

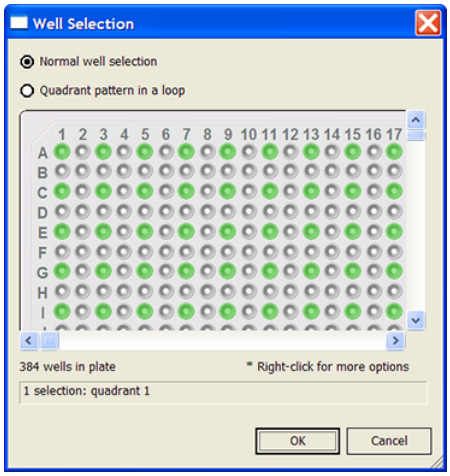
- The number of channels in the pipette head (or pins in a pin tool) is fewer than the number of wells in the microplate. For example, you can use a 96-channel pipette head to dispense liquid into a 384-well microplate or 1536-well microplate.
- All the channels are selected in the Set Head Mode task when using a pipette head. (The Set Head Mode task is not an option when using a pin tool).
- The liquid-handling task is inside a loop.

IMPORTANT

If you select a quadrant pattern, specifications in the Well Selection dialog box will override `task.Wellselection` values assigned in the Advanced Settings area.

To select a quadrant pattern:

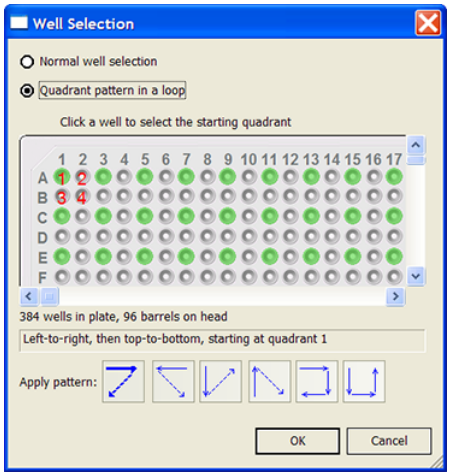
1 In the **Task Parameters** area, click the **Well selection** parameter box, and then click the Browse button. The Well Selection dialog box opens. By default, the **Normal well selection** option is selected. This option is used for column- and row-wise liquid-handling patterns.



2 Select **Quadrant pattern in a loop.** The contents of the dialog box change. Notice the following:

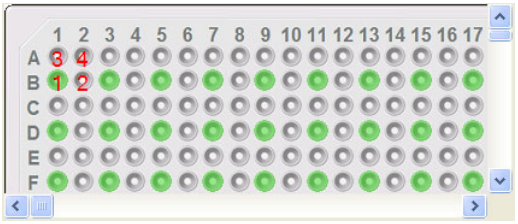
- Red numbers (1 through 4) appear on wells A1, A2, B1, and B2. The numbers indicate the pipetting sequence: 1 is the starting well, and 4 is the last well. In the following example, the sequence is A1, A2, B1, B2.
- Green wells indicate the starting well in the pipetting sequence.
- Pattern buttons at the bottom of the dialog box indicate the movement of the pipette channels. The movement description is provided in the text box above the buttons.

  Note: The last two patterns are unavailable if a group contains 16 wells. For example, the last two patterns are not available if you have a 96-well pipette head and a 1536-well microplate.



3 Select the starting well. The well becomes green and is labeled 1.

In the following example, the third quadrant (B1 well) is selected.



4 Click a pattern button to specify the pipette channel movement. After you click a pattern, the red numbers in the graphic are updated to show the sequence.

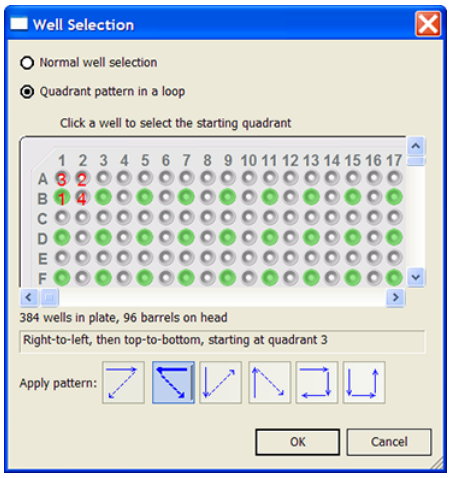
In the following example, the second pattern is selected (right-to-left, then top-to-bottom). The third quadrant (B1) is the starting well. The resulting movement is:

Quadrant 3 (B1)

Quadrant 2 (A2)

Quadrant 1 (A1)

Quadrant 4 (B2)



5 When you are finished, click **OK** to save the changes and return to the VWorks window.

Example: Allowing a form user to change aspirate flow rate using a form edit control

Goal

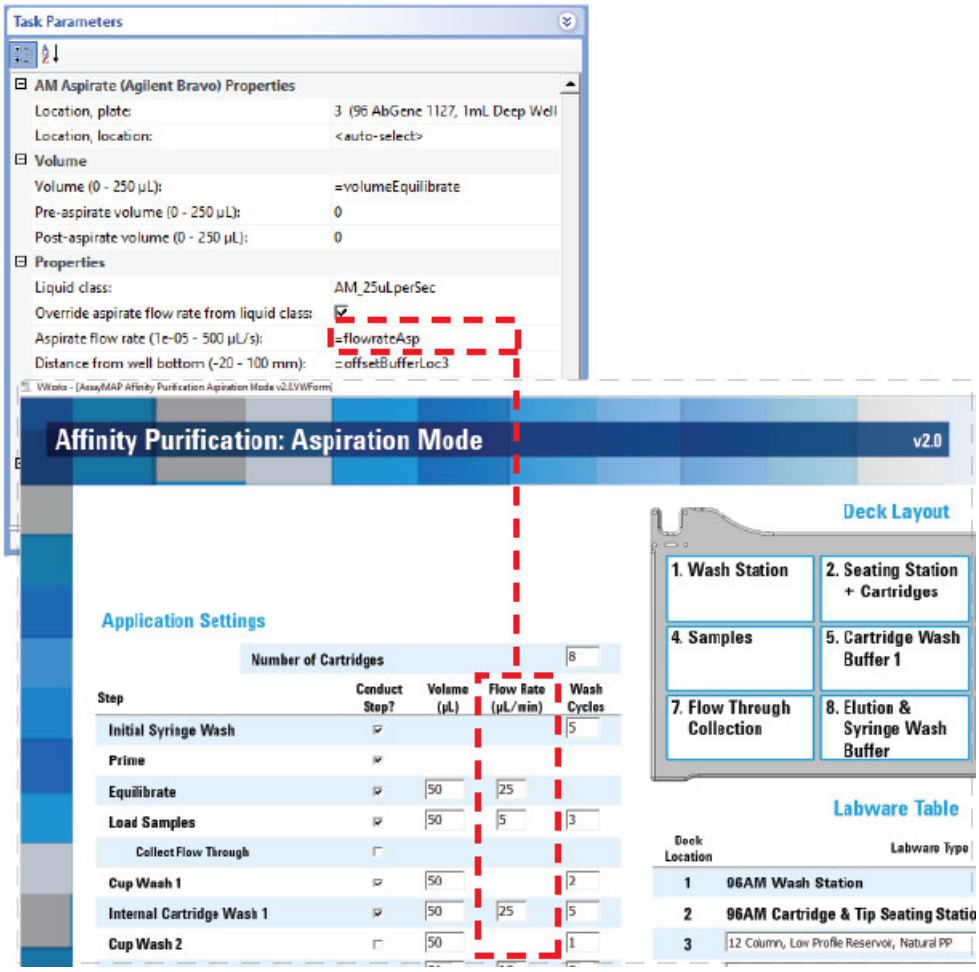
Allow a form's user to change the aspirate flow rate easily without modifying the liquid class.

Implementation

In the protocol, the AM Aspirate task is added. In the task parameters, the **Override aspirate flow rate from liquid class** check box is selected, and the **Aspirate flow rate** is specified as a variable.

In the VWorks form, an edit control is added for the aspirate flow rate using the `same` Aspirate flow rate variable. The form's user can enter values within a specified range in the form and run the protocol using the form. For details on how to design VWorks forms to run protocols, see "Creating forms for protocol operators" on page 177.

Figure Example of flow rate parameter variable and edit control in a VWorks form



Related information

For information about...	See...
Using the Bravo 96AM Head	*AssayMAP Bravo Platform Getting Started Guide*
AM Dispense task	"AM Dispense (Agilent Bravo)" on page 471
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514
AM Cartridges On	"AM Cartridges On (Agilent Bravo)" on page 491
AM Cartridges Off	"AM Cartridges Off (Agilent Bravo)" on page 489
Task macros	"Using macros to create protocols" on page 265

For information about...	See...
Protocol forms	"Creating forms for protocol operators" on page 177
Using JavaScript variables	"Using simple variables" on page 93

AM Dispense (Agilent Bravo)

Description

The AM Dispense (Agilent Bravo) (

) task is designed for dispensing fluids through AssayMAP Bravo cartridges that are mounted on the Bravo 96AM Head.

Task is available for...	Task is available in...
Bravo Platform with Bravo 96AM Head	Main Protocol, Bravo Subprocess

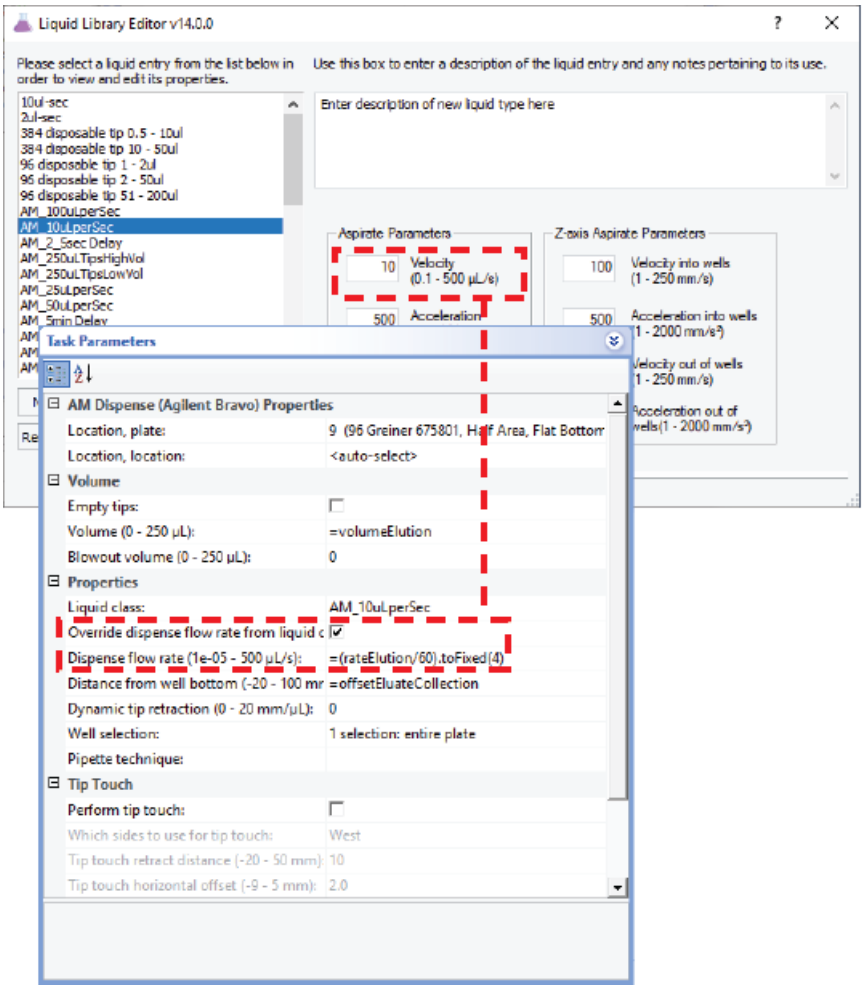
The AM Dispense (Bravo) task has the same task parameters as the Dispense (Bravo) task, except for the following parameters:

- Override dispense flow rate from liquid class

  Precise flow-rate control is critical for the performance of the AssayMAP Bravo cartridges. The option to override the dispense flow rate (velocity) in the liquid class enables an operator to change the flow rate from experiment to experiment for rapid method development without modifying the liquid class. Likewise, the flow rate can vary for different AM Dispense tasks that use the same liquid class in a given protocol.
- Distance from well bottom (-20–100 mm)

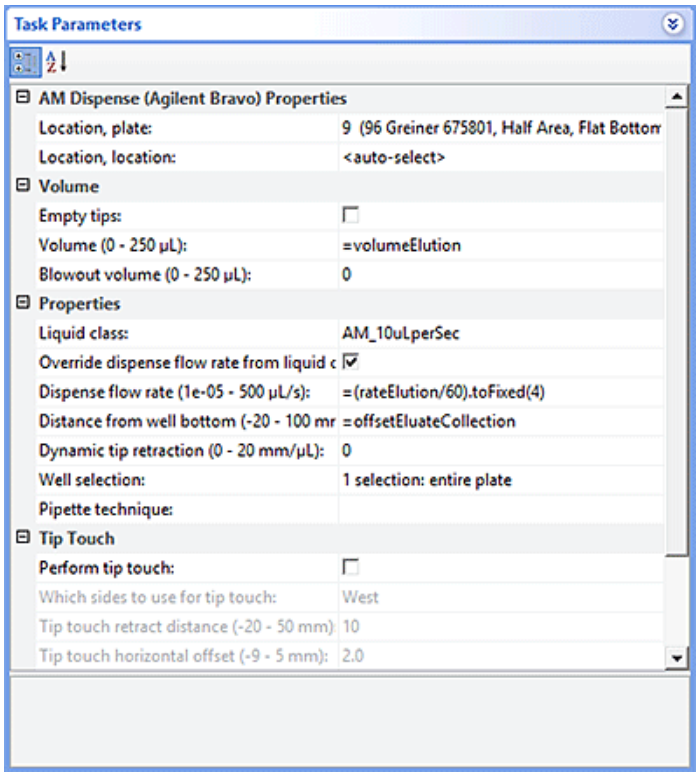
  The range includes negative values, which is critical for steps that require using the bare probes to dispense into the top portion of unmounted cartridges.

Figure Liquid class velocity setting in the Liquid Library Editor and flow rate override option



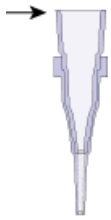
Task parameters

After adding the AM Dispense task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Empty tips	The option to empty all liquid from the probes, cartridges, or tips instead of using the dispense volume specification.
Volume (0–250 µL)	The volume of liquid to be dispensed from each probe, cartridge, or tip.
Blowout volume (0–250 µL)	Specifies the volume of air to dispense after the main volume has been dispensed while the probes, cartridges, or tips are still in the wells. Typically, the blowout volume is the same as the pre-aspirate volume. Note: Blowout occurs only in the last quadrant dispensed for a given AM Dispense task.
Liquid class	The pipetting speed and accuracy. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Override dispense flow rate from liquid class	The option to override the dispense velocity in the selected liquid class. Selecting this option enables you to specify a value for the dispense flow rate without changing the liquid class.

Parameter	Description
Dispense flow rate (1e05–500 µL/s)	The numerical value or the JavaScript variable that will override the dispense velocity setting in the liquid class. A JavaScript variable enables the value to be assigned later. For example, using a VWorks form, an operator could easily change the flow rate for an aspirate step in increments from as low as 1 µL/min up to 2000 µL/min or more using the same liquid class. If the task is included in a VWorks macro, a JavaScript variable enables you to change the value for the task at the macro level. **IMPORTANT** The software requires that the flow rate value be in microliters per second (µL/s) at run time. If you want an operator to enter the value in microliters per minute (µL/min), you can use scripting to convert the values for the software to use.

Parameter	Description
Distance from well bottom (-20–100 mm)	The distance between the tips of the probes, cartridges, or disposable tips and the *well bottoms* during the AM Dispense task. If you specify dynamic tip extension, this is the distance at the end of the AM Dispense task. Use a positive number for tasks that are performed with mounted cartridges, mounted tips, or bare probes at labware other than a cartridge rack. If the task is performed in the upper cup of unmounted cartridges, you can use a negative number for the parameter value. In this case, the *well bottom* is the top of the cartridge cup, as shown in the following figure.  For example, a negative number enables a prewetting step before mounting and priming the cartridges. **IMPORTANT** You can use a negative number for this parameter only if the AM Dispense task is performed in cartridges that are located in a cartridge rack. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the tips at the correct distance from the well bottom.

Parameter	Description
Dynamic tip retraction (0–20 mm/µL)	The rate at which to raise the pipette head during the Dispense task. Use dynamic tip retraction to prevent spills as the pipette tips displace the liquid. To move the tips: • *At the same rate as the volume change*. Calculate dynamic tip retraction (DTR) as follows: DTR = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTR > 1/A • *Slower than the volume change*. DTR < 1/A The starting and ending positions can be calculated as follows: $(V_{dispensed} * DTR) + Distance_{well\ bottom}$
Pipette technique	The pipette location offset you want to use for the AM Dispense task. The list of pipette techniques are defined in the Pipette Technique Editor.
Perform tip touch	The option to touch the probes, cartridges, or tips on one or more sides of the well.
Which sides to use for tip touch	The side or sides of the well to use during tip touch: North, South, East, West, North/South, West/East, West/East/South/North.
Tip touch retract distance (–20 to 50 mm)	The vertical distance for the probes, cartridges, or tips to rise before touching the sides of the wells.
Tip touch horizontal offset (–5 to 5 mm)	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. For example, if you set a value of: • 0, the tips move a horizontal distance equal to the well radius • > 0, the tips attempt to move past the well radius, which results in a more forceful tip touch • < 0, the tips move a distance less than the radius of the well, resulting in a lighter tip touch

Quadrant pattern well selection

A quadrant is an evenly spaced array of locations that are accessible by the tips on a pipette head.

Note: For the Bravo 96AM Head, the quadrant pattern option is available only if you are using 384-well microplates.

Instead of a column- or row-wise pattern, you can select a quadrant pattern during well selection.

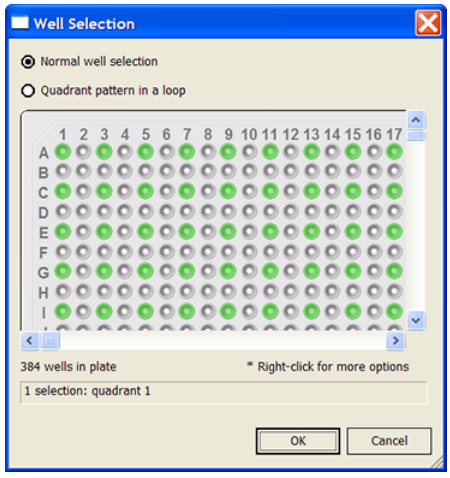
The quadrant pattern option is available only if:

- The number of channels in the pipette head (or pins in a pin tool) is fewer than the number of wells in the microplate. For example, you can use a 96-channel pipette head to dispense liquid into a 384-well microplate or 1536-well microplate.
- All the channels are selected in the Set Head Mode task when using a pipette head. (The Set Head Mode task is not an option when using a pin tool).
- The liquid-handling task is inside a loop.

IMPORTANT If you select a quadrant pattern, specifications in the Well Selection dialog box will override `task.Wellselection` values assigned in the Advanced Settings area.

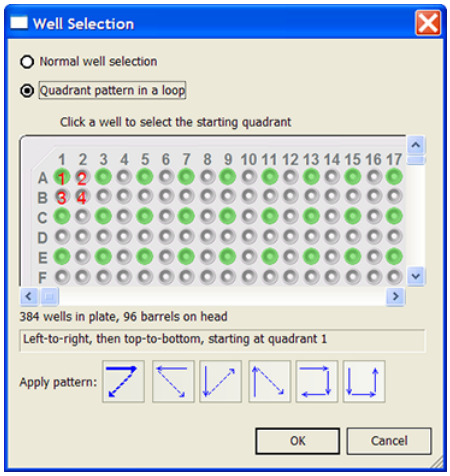
To select a quadrant pattern:

1. In the **Task Parameters** area, click the **Well selection** parameter box, and then click the Browse button. The Well Selection dialog box opens. By default, the **Normal well selection** option is selected. This option is used for column- and row-wise liquid-handling patterns.



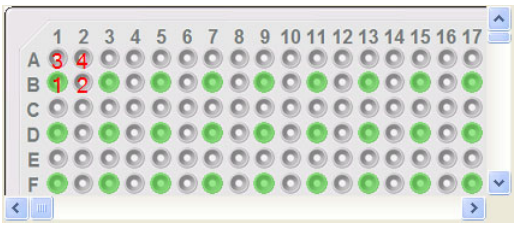
2. Select **Quadrant pattern in a loop.** The contents of the dialog box change. Notice the following:
   - Red numbers (1 through 4) appear on wells A1, A2, B1, and B2. The numbers indicate the pipetting sequence: 1 is the starting well, and 4 is the last well. In the following example, the sequence is A1, A2, B1, B2.
   - Green wells indicate the starting well in the pipetting sequence.
   - Pattern buttons at the bottom of the dialog box indicate the movement of the pipette channels. The movement description is provided in the text box above the buttons.

Note: The last two patterns are unavailable if a group contains 16 wells. For example, the last two patterns are not available if you have a 96-well pipette head and a 1536-well microplate.



3 Select the starting well. The well becomes green and is labeled 1.

In the following example, the third quadrant (B1 well) is selected.



4 Click a pattern button to specify the pipette channel movement. After you click a pattern, the red numbers in the graphic are updated to show the sequence.

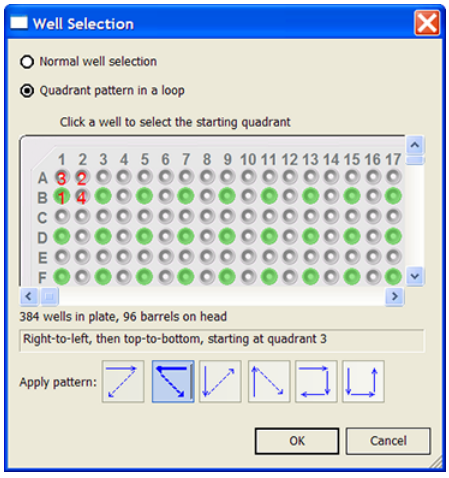
In the following example, the second pattern is selected (right-to-left, then top-to-bottom). The third quadrant (B1) is the starting well. The resulting movement is:

Quadrant 3 (B1)

Quadrant 2 (A2)

Quadrant 1 (A1)

Quadrant 4 (B2)



5 When you are finished, click **OK** to save the changes and return to the VWorks window.

Example: Allowing form's user to change dispense flow rate using a form edit control

Goal

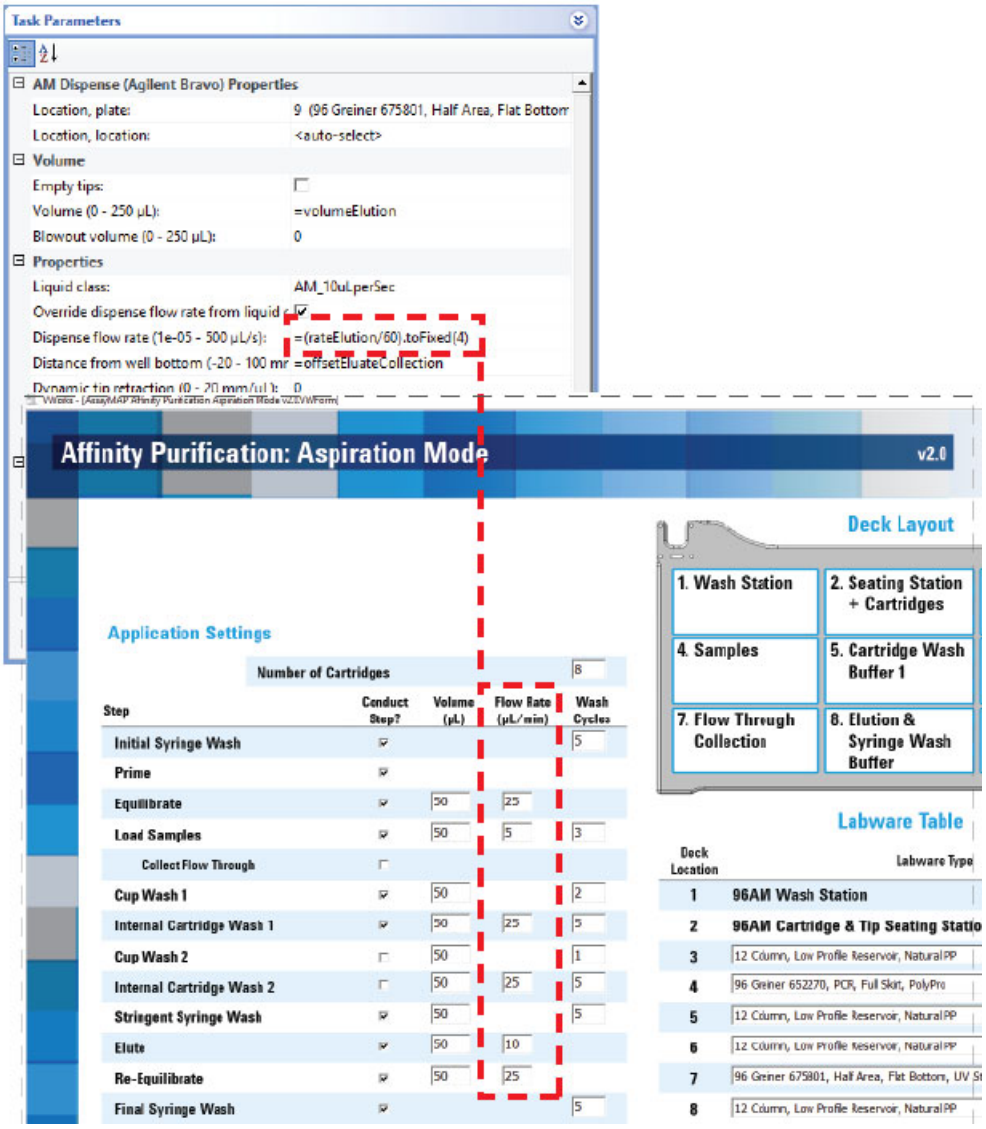
Allow a form's user to change the dispense flow rate without modifying the liquid class.

Implementation

In the protocol, the AM Dispense task is added. In the task parameters, the **Override dispense flow rate from liquid class** check box is selected, and the **Dispense flow rate** is specified as a variable.

In the VWorks form, an edit control is added for the dispense flow rate using the Dispense flow rate variable. The user can enter values within a specified range in the form and run the protocol using the form. For details on how to design VWorks forms to run protocols, see "Creating forms for protocol operators" on page 177.

Figure Flow rate parameter variable and edit control in a VWorks form



Related information

For information about...	See...
Using the Bravo 96AM Head	*AssayMAP Bravo Platform Getting Started Guide*
AM Aspirate task	"AM Aspirate (Agilent Bravo)" on page 461
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514
AM Cartridges On	"AM Cartridges On (Agilent Bravo)" on page 491

For information about...	See...
AM Cartridges Off	"AM Cartridges Off (Agilent Bravo)" on page 489
Task macros	"Using macros to create protocols" on page 265
Protocol forms	"Creating forms for protocol operators" on page 177
Using JavaScript variables	"Using simple variables" on page 93

AM Mix (Agilent Bravo)

Description

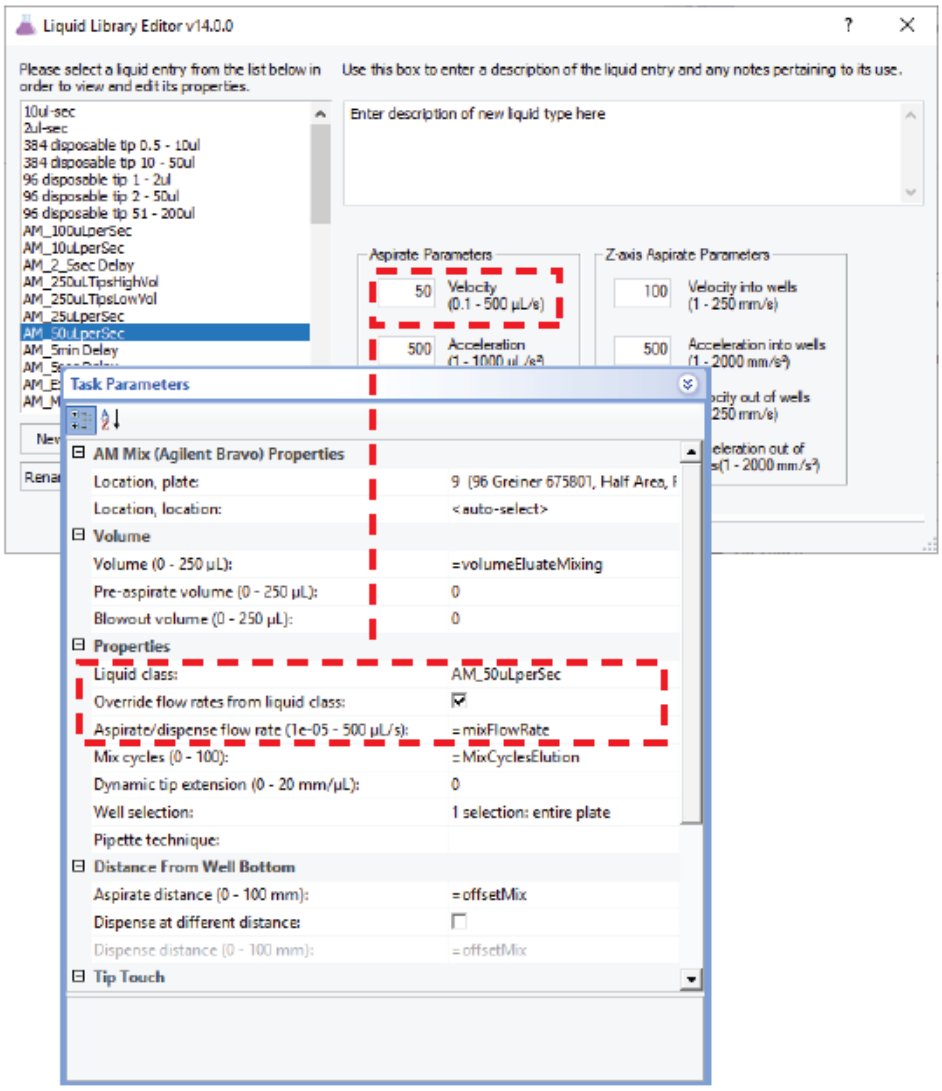
The AM Mix (Agilent Bravo) (

) task is designed for aspirating and dispensing fluids through AssayMAP Bravo cartridges that are mounted on the Bravo 96AM Head.

Task is available for...	Task is available in...
Bravo Platform with Bravo 96AM Head	Main Protocol, Bravo Subprocess

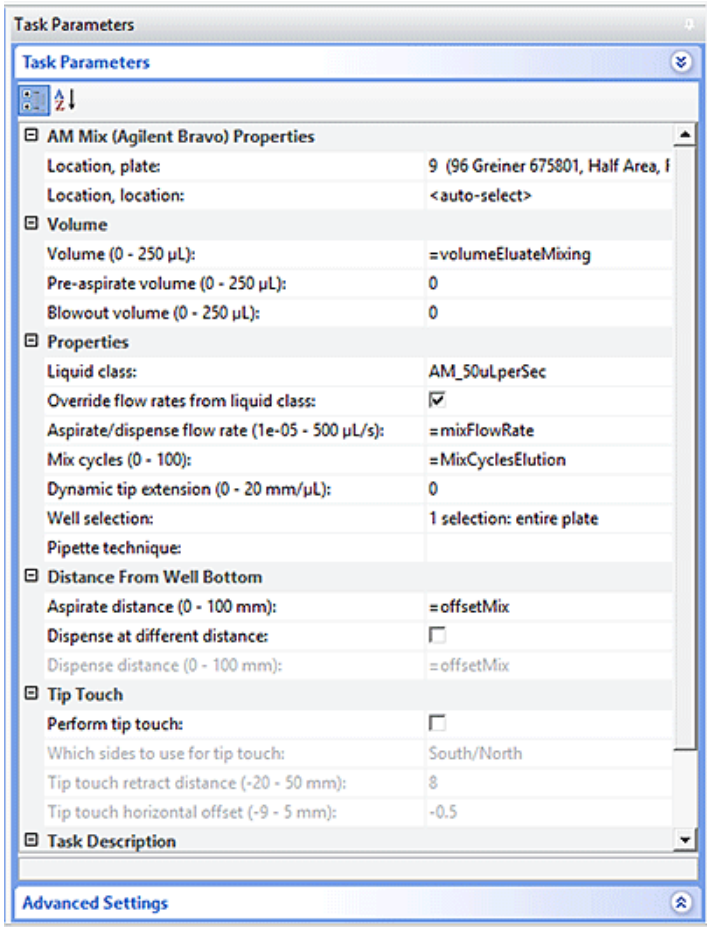
The AM Mix (Bravo) task has the same task parameters as the Mix (Bravo) task, plus an additional parameter that enables you to override the flow rate in the liquid class, if specified. Precise flow-rate control is critical for the performance of the AssayMAP Bravo cartridges. The option to override the aspirate and dispense flow rate (velocity) in the liquid class enables an operator to change the flow rate from experiment to experiment for rapid method development without modifying the liquid class. Likewise, the flow rate can vary for different AM Mix tasks that use the same liquid class in a given protocol.

Figure Liquid class velocity setting in the Liquid Library Editor and flow rate override option



Task parameters

After adding the AM Mix task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Volume (0–250 µL)	The volume of liquid to be mixed in each well.
Pre-aspirate volume (0–250 µL)	The volume of air to be drawn before the pipette tips enter the liquid.
Blowout volume (0–250 µL)	The volume of air to dispense after the main volume has been dispensed while the tips are still in the wells. Typically, the blowout volume is the same as the pre-aspirate volume.
Liquid class	The pipetting velocity and accuracy. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Override flow rates from liquid class	The option to override the aspirate and dispense velocity in the selected liquid class. Selecting this option enables you to specify a value for the flow rate without changing the liquid class.

Parameter	Description
Aspirate/dispense flow rate (1e-05–500 µL/s)	The numerical value or the JavaScript variable that will override both the aspirate and dispense velocity in the selected liquid class. A JavaScript variable enables the value to be assigned later. For example, using a VWorks form, an operator could easily change the flow rate for a mix step in increments from as low as 1 µL/min up to 2000 µL/min or more using the same liquid class. If the task is included in a VWorks macro, a JavaScript variable enables you to change the value for the task at the macro level. **IMPORTANT** The software requires that the flow rate value be in microliters per second (µL/s) at run time. If you want an operator to enter the value in microliters per minute (µL/min), you can use scripting to convert the values for the software to use.
Mix cycles (0–100)	The number of times to repeat the aspirate-and-dispense cycle.
Dynamic tip extension (0–20 mm/µL)	The rate at which the Bravo 96AM Head moves during the aspirate action. The software calculates the distance over which the tips will move without crashing. Use dynamic tip extension to prevent spills as the pipette tips displace the liquid. To move the tips: • *At the same rate as the volume change*. Calculate dynamic tip extension (DTE) as follows: DTE = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTE > 1/A • *Slower than the volume change*. DTE < 1/A The starting and ending positions can be calculated as follows: $(V_{aspirated} * DTE) + Distance_{well\ bottom}$ *Note:* Instead of a negative aspirated volume, the software automatically moves downward toward the well bottom with each aspirate action.

Parameter	Description
Pipette technique	The pipette location offset you want to use for the Mix task. The list of pipette techniques are defined in the Pipette Technique Editor.
Aspirate distance (0–100 mm)	The distance between the end of the pipette tips and the well botttoms during the aspirate action. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the tips at the correct distance from the well bottom.
Dispense at different distance	The option to dispense at a pipette tip height that is different than the aspirate distance. Select the check box to enter a value for the dispense distance.
Dispense distance (0–100 mm)	The distance between the tips of the probes, cartridges, or disposable tips and the well bottoms during the dispense action.
Perform tip touch	The option to touch the tips on one or more sides of the well.
Which sides to use for tip touch	The side or sides of the well to use during tip touch: North, South, East, West, North/South, West/East, West/East/South/North.
Tip touch retract distance (–20 to 50 mm)	The vertical distance for the tips to rise before touching the sides of the wells.
Tip touch horizontal offset (–5 to 5 mm)	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. For example, if you set a value of: • 0, the tips move a horizontal distance equal to the well radius • > 0, the tips attempt to move past the well radius, which results in a more forceful tip touch • < 0, the tips move a distance less than the radius of the well, resulting in a lighter tip touch

Quadrant pattern well selection

A quadrant is an evenly spaced array of locations that are accessible by the tips on a pipette head.

Note: For the Bravo 96AM Head, the quadrant pattern option is available only if you are using 384-well microplates.

Instead of a column- or row-wise pattern, you can select a quadrant pattern during well selection.

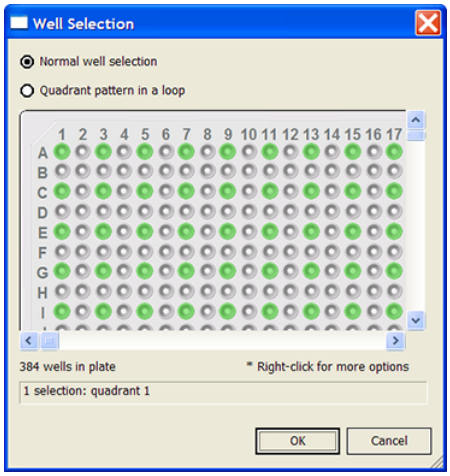
The quadrant pattern option is available only if:

- The number of channels in the pipette head (or pins in a pin tool) is fewer than the number of wells in the microplate. For example, you can use a 96-channel pipette head to dispense liquid into a 384-well microplate or 1536-well microplate.
- All the channels are selected in the Set Head Mode task when using a pipette head. (The Set Head Mode task is not an option when using a pin tool).
- The liquid-handling task is inside a loop.

IMPORTANT If you select a quadrant pattern, specifications in the Well Selection dialog box will override `task.Wellselection` values assigned in the Advanced Settings area.

To select a quadrant pattern:

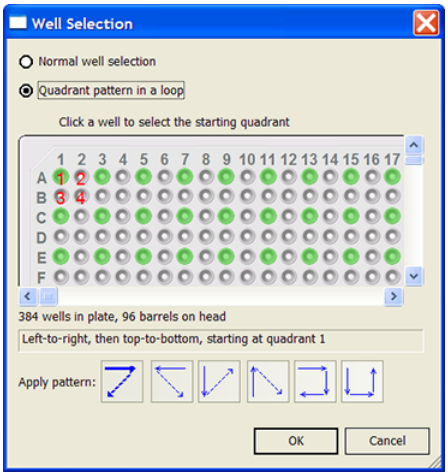
1 In the **Task Parameters** area, click the **Well selection** parameter box, and then click the Browse button. The Well Selection dialog box opens. By default, the **Normal well selection** option is selected. This option is used for column- and row-wise liquid-handling patterns.



2 Select **Quadrant pattern in a loop.** The contents of the dialog box change. Notice the following:

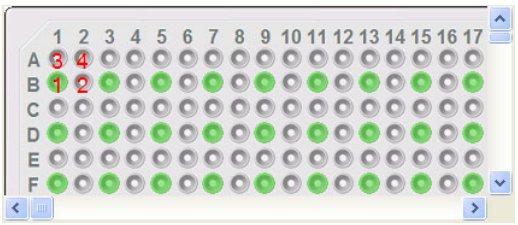
- Red numbers (1 through 4) appear on wells A1, A2, B1, and B2. The numbers indicate the pipetting sequence: 1 is the starting well, and 4 is the last well. In the following example, the sequence is A1, A2, B1, B2.
- Green wells indicate the starting well in the pipetting sequence.
- Pattern buttons at the bottom of the dialog box indicate the movement of the pipette channels. The movement description is provided in the text box above the buttons.

  Note: The last two patterns are unavailable if a group contains 16 wells. For example, the last two patterns are not available if you have a 96-well pipette head and a 1536-well microplate.



3 Select the starting well. The well becomes green and is labeled 1.

In the following example, the third quadrant (B1 well) is selected.



4 Click a pattern button to specify the pipette channel movement. After you click a pattern, the red numbers in the graphic are updated to show the sequence.

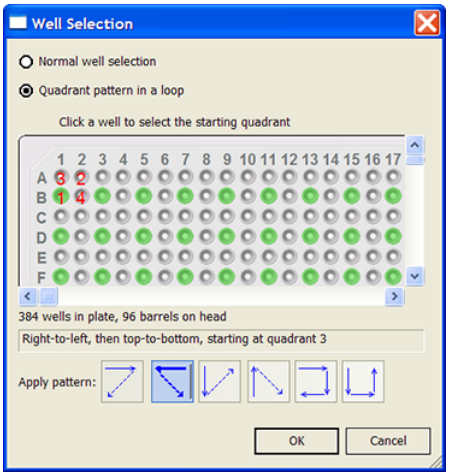
In the following example, the second pattern is selected (right-to-left, then top-to-bottom). The third quadrant (B1) is the starting well. The resulting movement is:

Quadrant 3 (B1)

Quadrant 2 (A2)

Quadrant 1 (A1)

Quadrant 4 (B2)



5 When you are finished, click **OK** to save the changes and return to the VWorks window.

Example: Allowing an operator to change the mix flow rate using an edit control on a VWorks form

Goal

Allow an operator to change the mix flow rate from run to run without modifying the liquid class.

Implementation

In the protocol, the AM Mix task is added. In the task parameters, the **Override dispense flow rates from liquid class** check box is selected, and the **Aspirate/dispense flow rate** is specified as a variable, for example `=mixFlowRate`.

In the VWorks form, an edit control is added for the mix flow rate using the `=mixFlowRate` variable. The operator can enter values within a specified range in the form and run the protocol using the form. For details on how to design VWorks forms to run protocols, see "Creating forms for protocol operators" on page 177.

Related information

For information about...	See...
Using the Bravo 96AM Head	*AssayMAP Bravo Platform Getting Started Guide*
AM Aspirate task	"AM Aspirate (Agilent Bravo)" on page 461
AM Dispense task	"AM Dispense (Agilent Bravo)" on page 471
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514
AM Cartridges On	"AM Cartridges On (Agilent Bravo)" on page 491
AM Cartridges Off	"AM Cartridges Off (Agilent Bravo)" on page 489
Task macros	"Using macros to create protocols" on page 265
Protocol forms	"Creating forms for protocol operators" on page 177
Using JavaScript variables	"Using simple variables" on page 93

AM Cartridges Off (Agilent Bravo)

Description

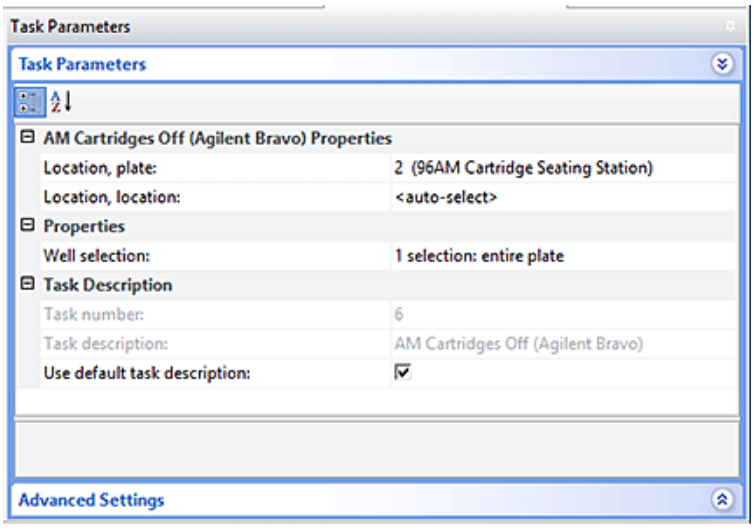
The AM Cartridges Off (Agilent Bravo) (

) task removes the AssayMAP Bravo cartridges from the Bravo 96AM Head. The task is available only if the device profile specifies the Bravo 96AM Head. You can use the AM Cartridges Off task while the head's syringes contain liquid, and then subsequently dispense the liquid using the syringe probes.

Task is available for...	Task is available in...
Bravo Platform with a gripper and a Bravo 96AM Head installed	Main Protocol, Bravo Subprocess

Task parameters

After adding the AM Cartridges Off task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The labware involved in the AM Cartridges Off task. Ensure a cartridge rack is selected.

Parameter	Description
Location, location	The location at which the AM Cartridges Off task occurs. <auto-select> automatically places the labware at the first-available or appropriate location for the task. If accessories are installed on the deck, the software uses the accessory configuration information in Bravo Diagnostics to determine the correct location for the task.
Well selection	The well positions to use for the AM Cartridges Off task. To select fewer positions than a full head, a Set Head Mode task must precede the AM Cartridges Off task in the protocol. **IMPORTANT** See the *AssayMAP Bravo Platform Getting Started Guide* for restrictions related to using a subset of channels in the Bravo 96AM Head.

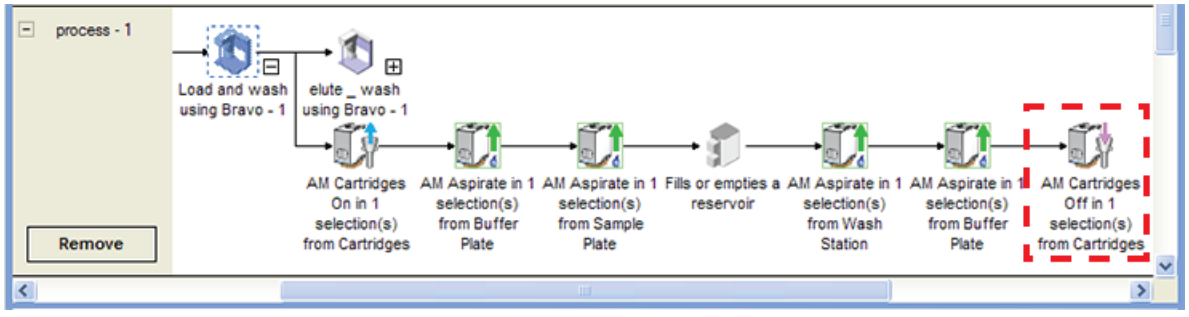
Example: Remove cartridges while the syringes contain liquid after a series of AM Aspirate tasks

Goal

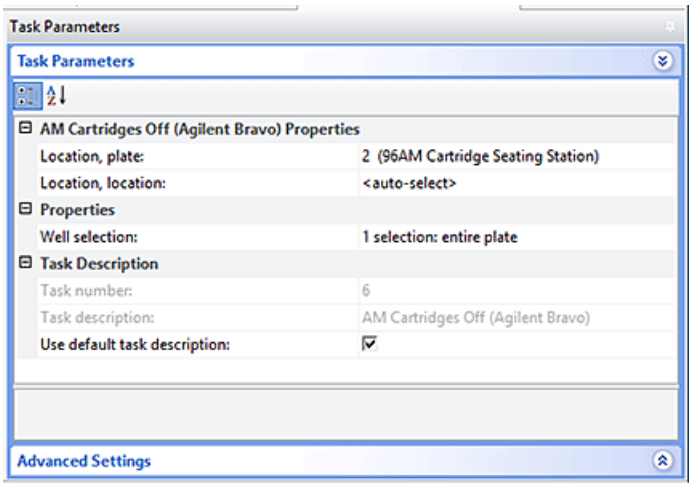
After aspirating a series of reagents through the AssayMAP Bravo cartridges to bind a target to the cartridge resin bed, remove the cartridges to wash the head's syringes before mounting the cartridges again for elution.

Implementation

In a liquid-handling subprocess, the AM Cartridges Off task is added after a series of AM Aspirate tasks and before a subsequent elute-wash subprocess.



The 96AM Cartridge Seating Station is selected as the Location, plate in the Task Parameters area.



Related information

For information about...	See...
Using the Bravo 96AM Head	*AssayMAP Bravo Platform Getting Started Guide*
AM Cartridges On task	"AM Cartridges On (Agilent Bravo)" on page 491
AM Aspirate task	"AM Aspirate (Agilent Bravo)" on page 461
AM Dispense task	"AM Dispense (Agilent Bravo)" on page 471
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Set Head Mode task	"Set Head Mode (Agilent Bravo)" on page 601

AM Cartridges On (Agilent Bravo)

Description

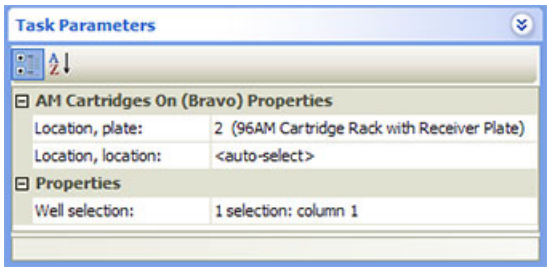
The AM Cartridges On (Agilent Bravo) (

) task presses cartridges on a Bravo 96AM Head. The task is available only if the profile specifies the Bravo 96AM Head. You can use the AM Cartridges On (Bravo) task while the head's syringes are empty or contain liquid.

Task is available for...	Task is available in...
Bravo Platform with a gripper and a Bravo 96AM Head installed	Main Protocol, Bravo Subprocess

Task parameters

After adding the AM Cartridges On task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The labware involved in the AM Cartridges On task. Ensure a cartridge rack is selected.
Location, location	The location at which the AM Cartridges On task occurs. <auto-select> automatically places the labware at the first-available or appropriate location for the task. If accessories are installed on the deck, the software uses the accessory configuration information in Bravo Diagnostics to determine the correct location for the task.
Well selection	The well positions to use for the AM Cartridges On task. To select fewer positions than a full head, a Set Head Mode task must precede the AM Cartridges On task. **IMPORTANT** See the *AssayMAP Bravo Platform Getting Started Guide* for restrictions related to using a subset of channels in the Bravo 96AM Head.

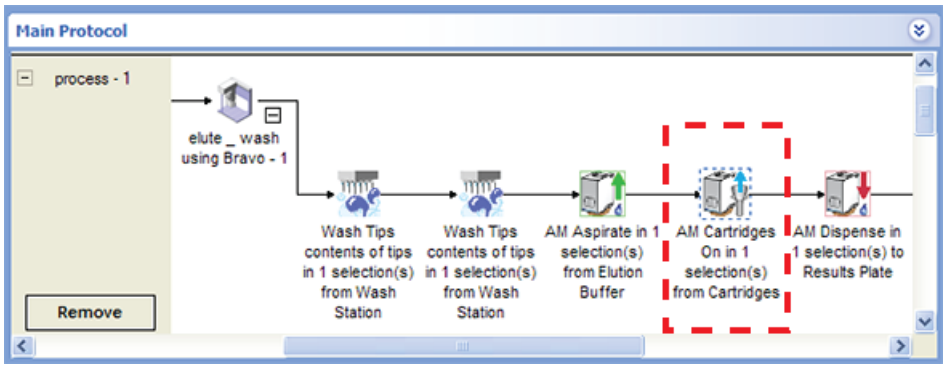
Example: Press on cartridges in order to perform a dispense through the cartridges

Goal

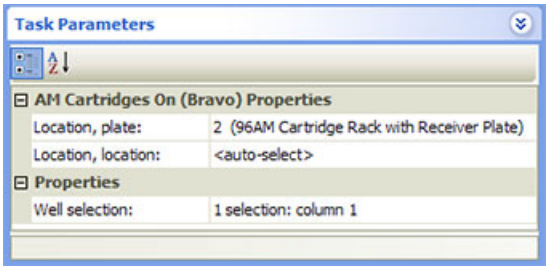
Press on the cartridges while fluid is held in the Bravo 96AM Head syringes so that the fluid can be dispensed through the cartridge resin bed for collection in a sample plate.

Implementation

In a liquid-handling subprocess, an AM Aspirate task is used to aspirate elution buffer using the bare probes into the Bravo 96AM Head syringes. The AssayMAP Bravo cartridges are pressed on using the AM Cartridges On task. An AM Dispense task is used to dispense the elution buffer through the cartridges into a results plate.



Because the cartridges are pressed on at a cartridge rack, the Cartridge Rack is selected for the Location, plate in the Task Parameters area.

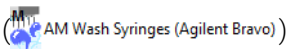


Related information

For information about...	See...
Using the Bravo 96AM Head	*AssayMAP Bravo Platform Getting Started Guide*
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
AM Cartridges Off task	"AM Cartridges Off (Agilent Bravo)" on page 489
AM Aspirate task	"AM Aspirate (Agilent Bravo)" on page 461
AM Dispense task	"AM Dispense (Agilent Bravo)" on page 471
Set Head Mode task	"Set Head Mode (Agilent Bravo)" on page 601

AM Wash Syringes (Agilent Bravo)

Description

The AM Wash Syringes (Agilent Bravo) (

) task enables the following types of washes using the Bravo 96AM Head:

- Internal washing of the Bravo 96AM Head syringes
- External washing of the syringe probes
- External washing of the AssayMAP Bravo cartridges while mounted on the probes

Task is available for...	Task is available in...
Bravo Platform with a Bravo 96AM Head and an autofilling station	Main Protocol, Bravo Subprocess

Requirements

In Bravo Diagnostics, ensure that the 96AM Wash Station is set up at the correct deck location and associated with a Pump Module.

Ensure that the protocol has a configured labware for the 96AM Wash Station in addition to the other processes and configured labware required by the protocol.

Task parameters

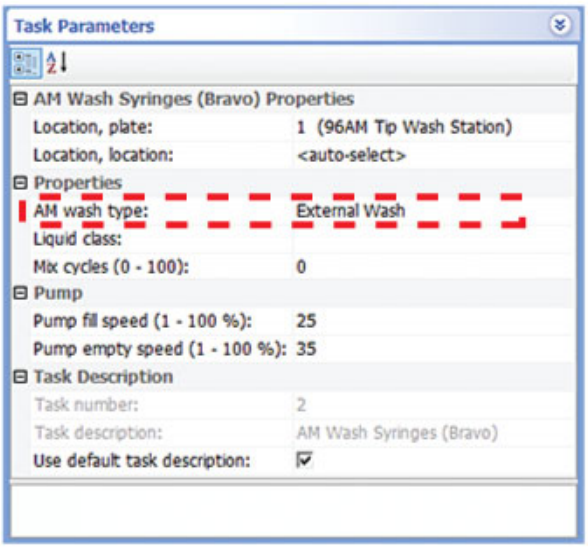
After adding the AM Wash Syringes task at the desired point in the protocol, you set the parameters in the **Task Parameters** area.

CAUTION **Ensure that all the syringes have bare probes or that all syringes have mounted cartridges for this task. Otherwise, a potential collision can occur, which can damage the syringes and the 96AM Wash Station.**

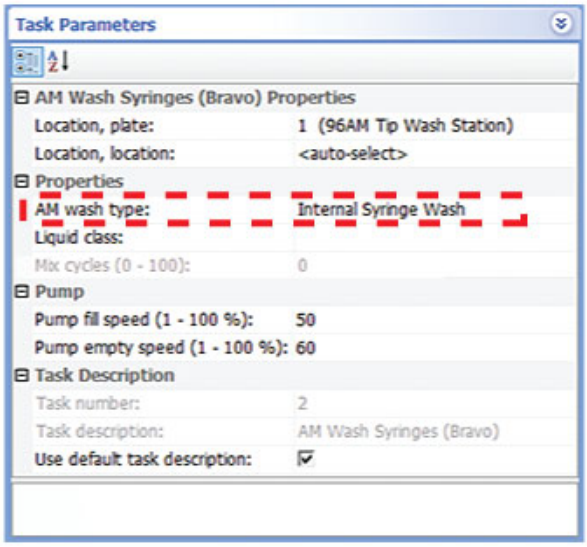
CAUTION **Ensure that the labware definition for the 96AM Wash Station is accurate and the teachpoint for the location is precise so that the system can position the syringe probes or cartridge tips at the correct location. Otherwise, a potential collision can occur, which can damage the syringes and the 96AM Wash Station.**

The software automatically selects all the chimneys in the wash station (entire plate) for the well selection and sets the volume and other properties. The following figure shows the **AM wash type** parameter: (**A**) **External Wash** or (**B**) **Internal Syringe Wash**.

A



B



The task actions vary depending on what you select for the **AM wash type** parameter:

- **External Wash**. Moves the tips of the probes or cartridges into the 96AM Wash Station chimneys. This wash type does not aspirate or dispense any volume. The liquid class can be used to control the duration of the wash.

  Select the **External Wash** to wash the tips of the syringe probes or the cartridges mounted on the probes. See "Example External Wash" on page 497.

- **Internal Syringe Wash**. Dispenses to waste any contents remaining in the syringes, and then aspirates 250-µL wash liquid from the 96AM Wash Station chimneys into the syringes. The task dispenses the 250-µL volume to waste outside the chimneys, and then performs a tip touch on the chimneys' exterior.

  Select the **Internal Syringe Wash** to flush the syringe barrels with wash liquid. See "Example Internal Syringe Wash" on page 497.

CAUTION

Ensure that all syringe probes are bare (cartridges are off) for the Internal Syringe Wash. Otherwise, a potential collision can occur, resulting in equipment damage.

For each wash you select the following parameters.

Parameter	Description
AM wash type	The type of wash: External Wash (default) or Internal Syringe Wash.

Parameter	Description
Liquid class	The liquid class for this wash liquid. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks. *Note:* Although the External Wash does not include aspirate and dispense actions, you can use a liquid class with a delay to control the duration of the wash.
Mix cycles (0–100)	*Internal Syringe Wash only.* The number of times to repeat the aspirate-and-dispense cycle for the wash task. Each mix cycle consists of one aspirate action and one dispense action.
Pump fill speed (%)	The speed, in percent of maximum, at which liquid flows into the 96AM Wash Station inlet ports and up into the chimneys. The Pump fill speed should be fast enough for the wash liquid to just bubble over the tops of the chimneys. Initially, you should use the following values, and then determine if you need to make adjustments: • **External Wash** – Pump fill speed 25% – Pump empty speed 35% • **Internal Syringe Wash** – Pump fill speed 50% – Pump empty speed 60%
Pump empty speed (%)	The speed, in percent of maximum, at which the waste liquid is pumped out of the 96AM Wash Station outlet ports. You should use the default values initially, and then determine if you need to adjust the value. **CAUTION** **The Pump empty speed should be slightly faster than the Pump fill speed to prevent an overflow.**
Use default task description	The option to change the label that appears with the task icon in the Main Protocol area. • To use the default text, ensure that the check box is selected. • To create your own label, clear the check box, and then type the desired text in the field next to **Task description**.

Example External Wash

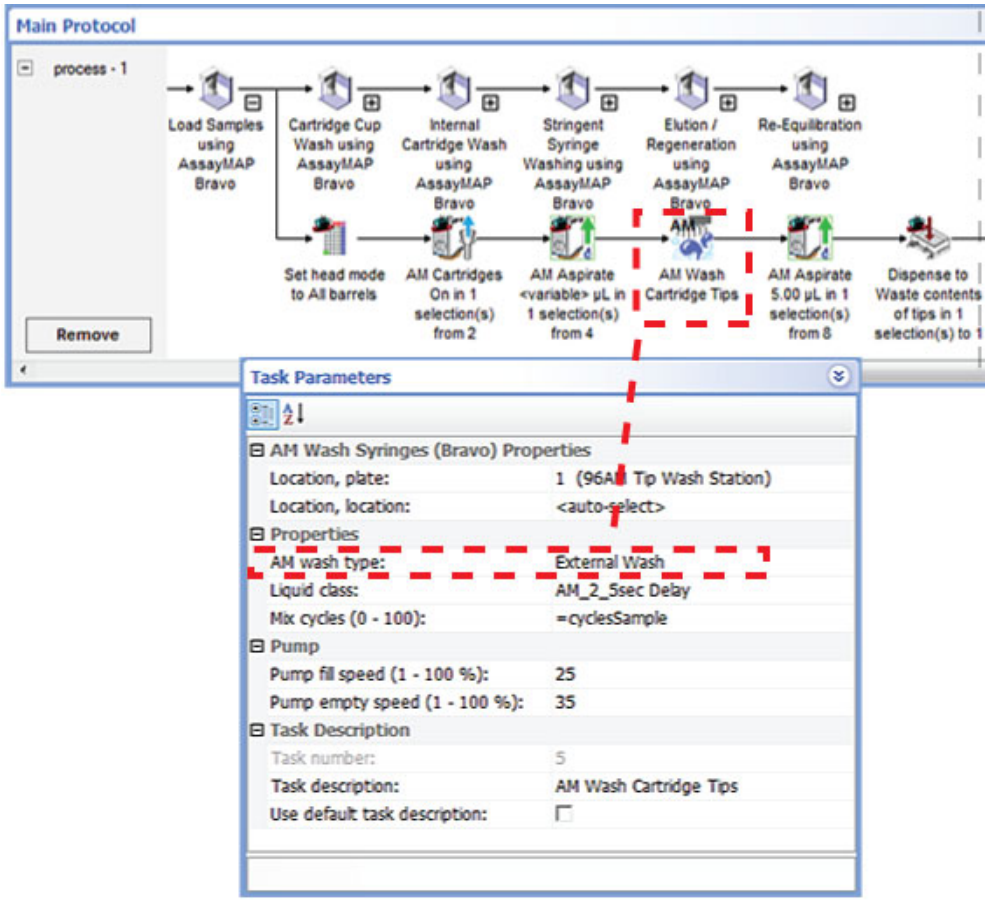
Goal

After aspirating samples or other reagents through the AssayMAP cartridges, perform an External Wash to remove the residual liquid that can cling to the external surfaces of the cartridges and prevent carryover to future steps.

Implementation

In the following example, the configured wash station at deck location 1 is called 96AM Tip Wash Station. In the Load Samples subprocess, an External Wash (AM Wash Cartridge Tips) task is added between an AM Aspirate task to load the sample on the mounted cartridges and an AM Aspirate task that aspirates 5 µL buffer as a sample chase. In the External Wash parameter settings, the liquid class specifies a delay that defines the time duration that the tips remain in the chimneys.

Figure External Wash example in an AssayMAP subprocess to load samples



Example Internal Syringe Wash

Goal

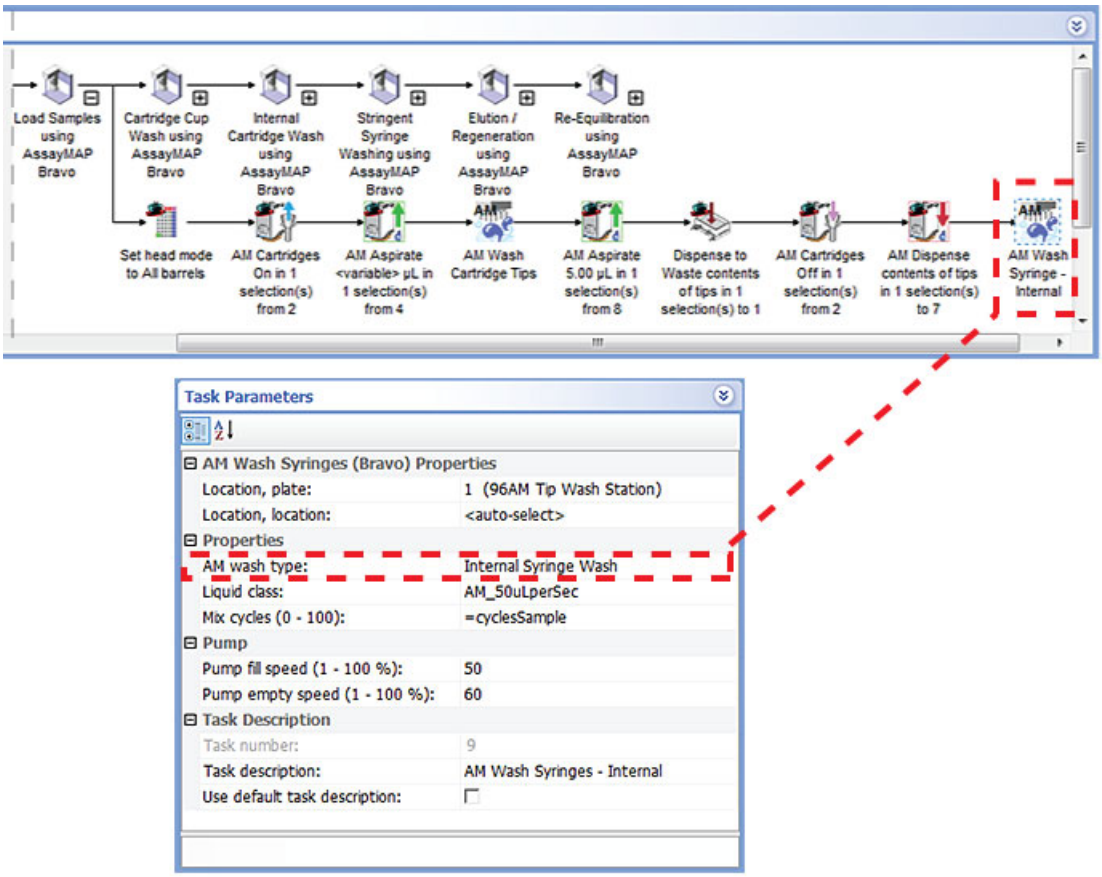
After aspirating samples or other reagents through the AssayMAP cartridges and into the syringes, eject the cartridges, and perform an Internal Syringe Wash to remove unwanted residual reagents from the syringes.

Implementation

In the following example, the configured wash station at deck location 1 is called 96AM Tip Wash Station. In the Load Samples subprocess, an Internal Syringe Wash task is added at the end of the subprocess after the cartridges are removed at deck location 2 and the syringe contents have been dispensed for collection at deck location 7.

Note: The Internal Syringe Wash dispenses to waste any contents remaining in the syringes when the task begins.

Figure Internal Syringe Wash example in an AssayMAP subprocess to load samples



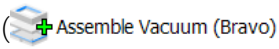
Related information

For information about...	See...
Using the Bravo 96AM Head	*AssayMAP Bravo Platform Getting Started Guide*
AM Aspirate task	"AM Aspirate (Agilent Bravo)" on page 461
AM Dispense task	"AM Dispense (Agilent Bravo)" on page 471
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514

For information about...	See...
AM Cartridges On	"AM Cartridges On (Agilent Bravo)" on page 491
AM Cartridges Off	"AM Cartridges Off (Agilent Bravo)" on page 489
Task macros	"Using macros to create protocols" on page 265
Protocol forms	"Creating forms for protocol operators" on page 177
Using JavaScript variables	"Using simple variables" on page 93

Assemble Vacuum (Agilent Bravo)

Description

The Assemble Vacuum task (

) directs the robot to pick up the Vacuum Filtration Station components from designated deck locations and stack them in the order you specify.

The task is used in conjunction with the following tasks:

- Disassemble Vacuum
- Move and Filter Plate
- Pipette Under Vacuum Collar
- Toggle Vacuum

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Supported configurations

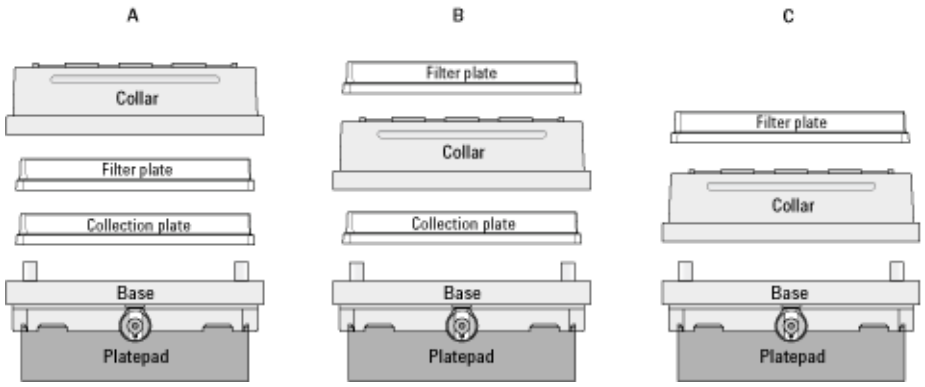
To accommodate different assay types, the Bravo Platform supports the Vacuum Filtration Station configurations shown in the following figure. Note the following:

- *Configuration A*. The filter plate is part of the station assembly. The assembly process can be automated during the protocol run. The filtrate in the collection plate is retained.

  Note: To perform pipetting in the configuration A filter plate, the pipetting task must be enclosed with a pair of Pipette Under Vacuum Collar tasks. For details, see "Pipette Under Collar Begin and End (Agilent Bravo)" on page 584.

  Note: Double-filtration is possible in configuration A by including one filter plate as part of the station assembly and then moving a second filter plate on top of the vacuum collar after the assembly is finished.

- *Configuration B*. The filter plate is not part of the station assembly. The assembly process can be automated during the protocol run. The robot will move the filter plate to the station after the assembly process is finished. The filtrate in the collection plate is retained.
- *Configuration C*. The filter plate is not part of the station assembly. The assembly process can be automated during the protocol run. The robot will move the filter plate to the station after the assembly process is finished. The filtrate is discarded.

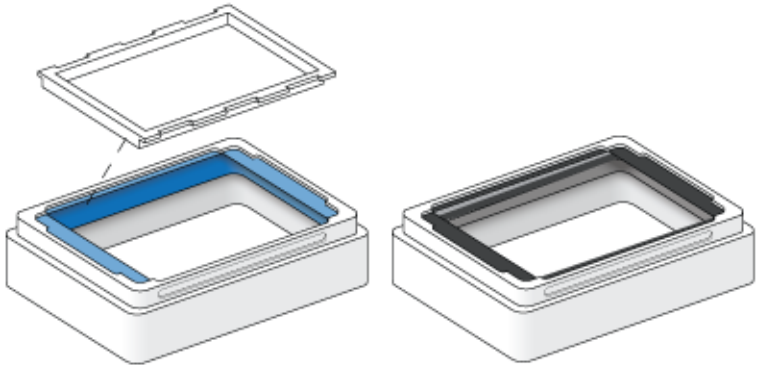


IMPORTANT

Ensure that the base of the station is installed on a short platepad (SRT platepad). If the Vacuum Filtration Station configuration is too tall for the Bravo gripper to assemble without the risk of a collision, an error will occur during the run.

If the type of filter plate cannot be seated in the clear frame support that sits in the blue gasket of the collar (configuration B or C), you may use black gaskets in the collar instead of the blue gasket and frame support. In this case, ensure one black gasket adheres to the top and one black gasket adheres to the underside of the collar.

Figure Collar options: blue gasket and gasket frame support or black gaskets



Note: If you use the black gasket to accommodate a filter plate, but the filter plate nozzles are too short to reach the wells in the collection plate underneath the collar, you may omit the black gasket on the underside of the collar. In this case, the Bravo device profile must specify the configuration as **Black gasket on top side only**.

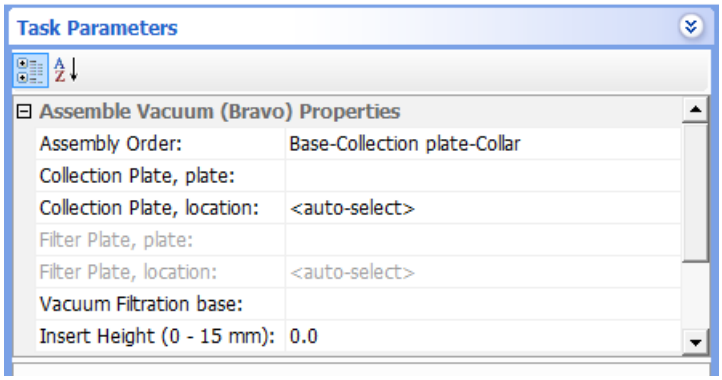
Requirements

To use the Assemble Vacuum (Bravo) task, you must first:

- Install the Vacuum Filtration Station, and then configure the accessory in Bravo Diagnostics as described in the *Bravo Platform User Guide*.
- In the VWorks Labware Editor, do the following:
  - Set the Robot gripper offset parameter for labware that will be placed on the Vacuum Filtration Station during a protocol run.
  - Define the filter plate parameters for length of filter tip and the filter-channel resting depth.

Task parameters

After adding the Assemble Vacuum (Bravo) task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Assembly order	The order, from bottom to top, in which to stack the station components. The selections are: • Base–Collection plate–Filter plate–Collar (configuration A) • Base–Collection plate–Collar (configuration B) • Base–Collar (configuration C) **IMPORTANT** The Vacuum Filtration Station collar must match the type (standard or deep) and the gasket configuration specified in the Bravo device profile.
Collection Plate, plate	*Available for configurations A and B only*. The process lane in the protocol for collecting the filtrate. Select the collection plate from the list. Depending on how the protocol is authored, the collection plate could be a plate process or configured labware.

Parameter	Description
Collection Plate, location	*Available for configurations A and B only*. The location of the collection plate. Select the deck location of the collection plate. Select <auto-select> only if the software knows of the collection plate location (for example, if you used a Place Plate task earlier in the protocol to specify its location).
Filter Plate, plate	*Available for configuration A only*. The labware to use as the filter plate. Select the filter plate from the list. Depending on how the protocol is authored, the collection plate could be a plate process or configured labware.
Filter Plate, location	*Available for configuration A only*. The location of the filter plate. Select the deck location of the filter plate. Select <auto-select> only if the software knows of the filter plate location (for example, if you used a Place Plate task earlier in the protocol to specify its location).
Vacuum Filtration base	The deck location of the Vacuum Filtration Station base.
Insert height (0–15 mm)	The height of the white plastic spacer, if present, that can be placed in the base manually before starting the protocol. The insert is used to raise the height of the collection plate and reduce the spacing between the filter plate nozzles and the collection plate underneath. Measure the height (mm) from the bottom edge to the top edge of the insert. If you are using multiple inserts, measure the combined height of the stack of inserts.  If no insert is in the base, use a value of 0.0 mm (default). **CAUTION** If the assembled stack of plates and inserts is too tall, the collar of the Vacuum Filtration Station may not seal properly during the vacuum tasks, and an error message will appear.

Examples:

See the example in:

- "Move and Filter Plate (Agilent Bravo)" on page 568
- "Toggle Vacuum (Agilent Bravo)" on page 626

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • *Bravo Platform User Guide*
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Configuring the Vacuum Filtration Station	*Bravo Platform User Guide*
Setting the robot gripper offset in the Labware Editor	*VWorks Automation Control Setup Guide*
Move and Filter Plate task	"Move and Filter Plate (Agilent Bravo)" on page 568
Pipetting in a filter plate that sits under the collar	"Pipette Under Collar Begin and End (Agilent Bravo)" on page 584
Toggle task	"Toggle Vacuum (Agilent Bravo)" on page 626
Disassemble Vacuum (Bravo) task	"Disassemble Vacuum (Agilent Bravo)" on page 512
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Dilute to Final Volume (Agilent Bravo)

Description

The Dilute to Final Volume (Agilent Bravo) task (

) allows you to transfer liquid from a reagent labware to a destination microplate. The task is available only if you specified dilution series parameters within the Format Wizard in the Hit Pick Replication task.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

You can add the Dilute to Final Volume task before or after the Hit Pick Replication task.

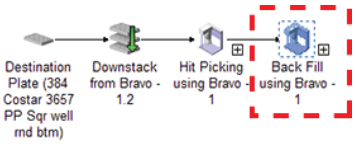
- *Before hit-picking*. Prefills the destination microplate with reagent. If the destination microplates are empty when the task starts, you only need to change the pipette tip with every destination microplate. You do not have to change the pipette tip between each reagent well, thus reducing the number of tips used.
- *After hit-picking*. Backfills the destination microplate with reagent. In this case, you must change the pipette tip between each reagent well to prevent contamination, thus using more tips.

Adding the Dilute to Final Volume task

To add the Dilute to Final Volume task:

1 Add another Bravo Subprocess in the destination microplate process. (You cannot add the Dilute to Final Volume task in the subprocess that contains the Hit Pick Replication task.)

In the following example, the first Bravo subprocess (called Hit Picking) contains the Hit Pick Replication task. The second Bravo subprocess, Back Fill, contains the Dilute to Final Volume task. Notice that both subprocesses are in the Destination Plate process.



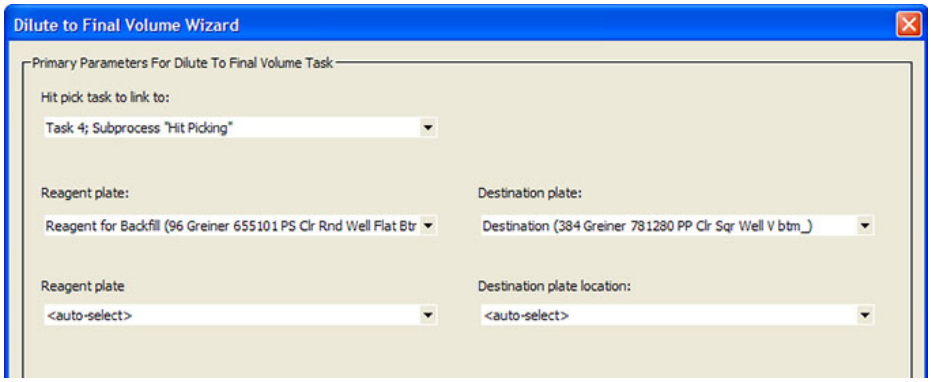
When you add the Dilute to Final Volume () task, the software automatically starts the Dilute to Final Volume Wizard.

2 On the first page of the wizard:

a Select the source and destination labware to use.

Note: The source labware is the labware that contains the reagent for back filling. It is not the source microplate used for the Hit Pick Replication task.

b Specify the locations of the reagent and destination labware.



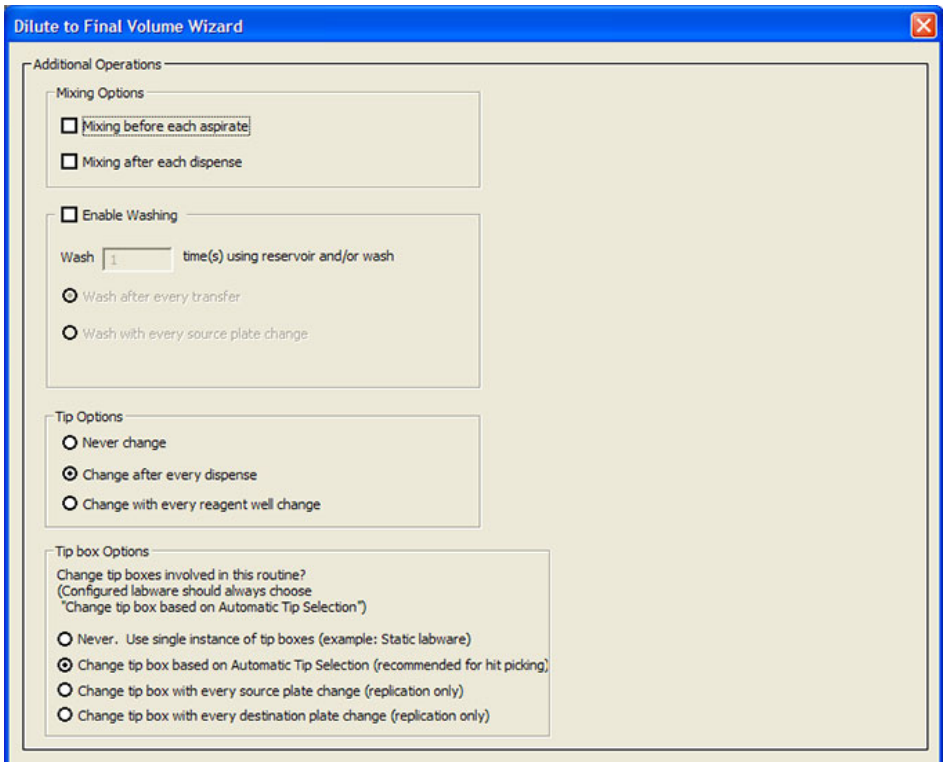
When you are finished, click **Next.**

3 Specify the Aspirate and Dispense task parameters. For detailed parameter descriptions, see "Aspirate (Agilent Bravo)" on page 455 and "Dispense (Agilent Bravo)" on page 514.

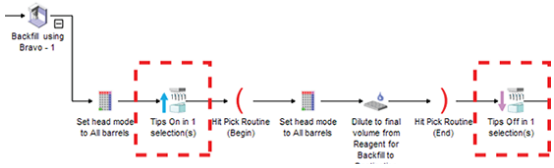


When you are finished, click **Next.**

4 Select additional operations, if applicable.

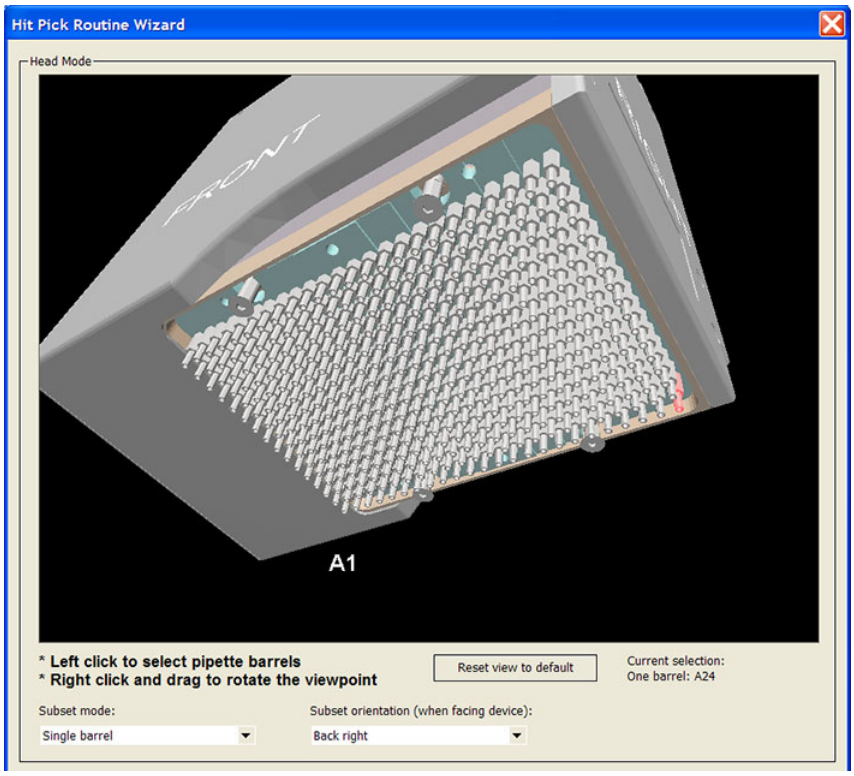


Additional operation	Description
Mixing Options	Select where you want to add a Mix task: • Before each aspirate task • After each dispense task You will need to set the Mix parameters later in the wizard. If you do not want to mix, clear both check boxes.
Enable Washing	Select the check box if you want to add one or more Wash Tips tasks. You can specify the number of tip washes and whether you want to: • Wash after every transfer • Wash with every source microplate change You will need to set the Wash Tips parameters later in the wizard. Clear the check box if you do not want to add any Wash Tips tasks.

Additional operation	Description
Tip Options	Select the tip-change option: • *Never change*. If you select option, you must add the Set Head Mode, Tips On, and Tips Off tasks in the subprocess, outside of the Dilute to Final Volume routine, after you are finished with the wizard.  • *Change after every dispense*. Changes the tips after every dispense within the routine. • *Change with every reagent well change*. Changes the tips after every reagent well change during the routine.
Tip Box Options	Select when to change the tip box: • *Never. Use single instance of tip boxes (example: Static labware)*. Does not change the tip box within the routine. • *Change tip box based on Automatic Tip Selection*. Uses tip-tracking within the routine, and changes the tip box when the box if fully used. Select this option if the tip boxes you are using are configured labware. *Note:* Tip-tracking continues in subsequent tasks in the process. • *Change tip box with every source plate change (replication only)*. Changes the tip box every time the source microplate changes. This option is available only when a replication format file is selected. • *Change tip box with every destination plate change (replication only)*. Changes the tip box every time the destination microplate changes. This option is available only when a replication format file is selected.

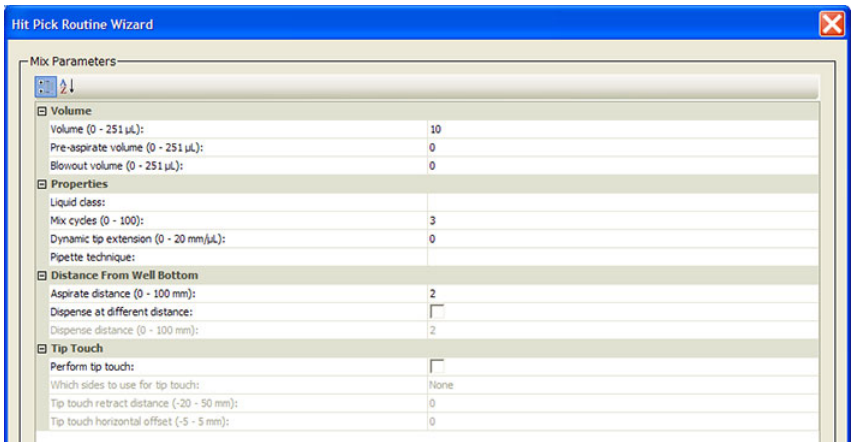
When you are finished, click **Next.**

5 Select the pipette channels to use. For a description of the selections, see "Set Head Mode (Agilent Bravo)" on page 601.



When you are finished, click **Next.**

6 *If you selected mixing options in step 4*. Set the mixing parameters. For a description of the parameters, see "Mix (Agilent Bravo)" on page 559.



When you are finished, click **Next.**

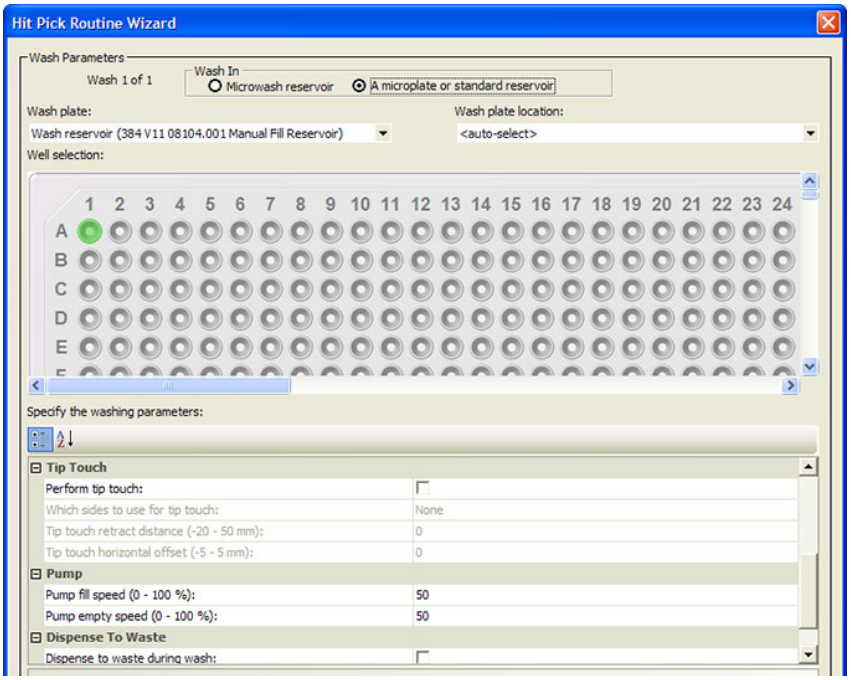
7 *If you selected wash options in step 4*. Set the wash parameters.

a In the **Wash In** area, select the type of labware to be used for the wash task:

- *MicroWash Reservoir (MicroWash Reservoir)*. Adds Wash Tips tasks and will use either configured or static labware.
- *A microplate or standard reservoir*. Adds Mix tasks.

b In the **Wash plate** list, select the desired labware and location.

c Set wash-tip parameters. For a description of the parameters, see "Wash Tips (Agilent Bravo)" on page 632.



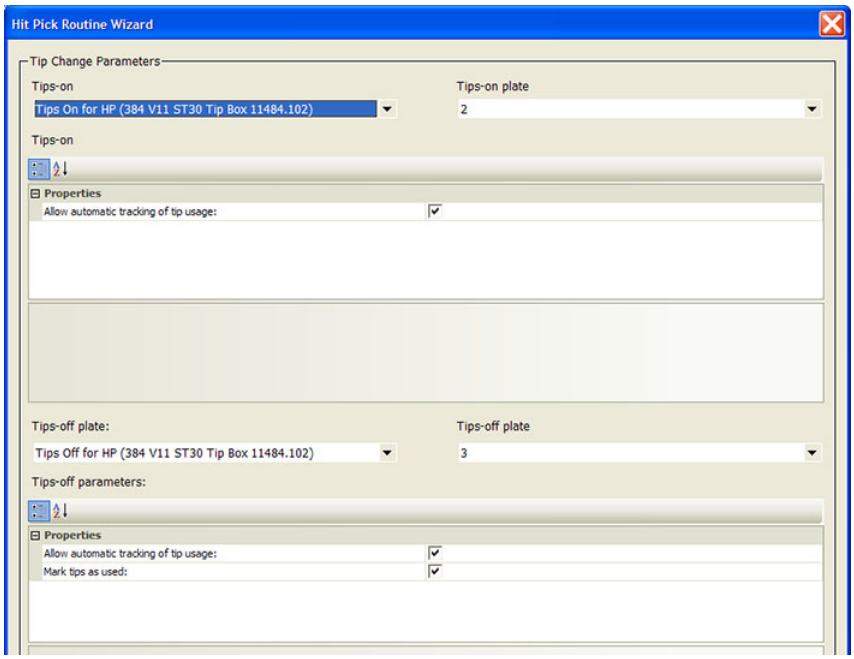
When you are finished, click **Next.**

8 *If you selected a tip-change option in step 4*. Select the tip box options.

a Select the tip boxes you want to use.

b Select the tip-tracking options you want to use:

- *Allow automatic tracking of tip usage*. Allows the software to track pipette tip usage during the protocol run or across different protocol runs. If you select the option in this task, you must also select the option in the Tips Off task. In general, use the default selections displayed.
- *Mark tips as used*. The option to use only new pipette tips during the protocol run. Select the option so that the software counts the number of tips used during the protocol run. The tips that have been used once are marked as used so that they cannot be reused. Clear the check box so that during the next Tips On task, the same tips can be reused. In general, use the default selections displayed.



9 Click **Finish.**

Based on the information you provide in the wizard, the software adds the Aspirate, Dispense, Mix, tip-washing, and other tasks that are necessary to produce the desired results.

Notice the following:

- The Hit Pick Routine (Begin) and Hit Pick Routine (End) tasks mark the beginning and end of the dilute-to-final-volume tasks.
- You can modify any of the task parameters directly in the Task Parameters area.
- You can return to the Dilute to Final Volume Wizard. To do this, select either the Hit Pick Routine (Begin) or Hit Pick Routine (End) task. In the Task Parameters area, click **Launch hit pick routine wizard.**

Related information

For information about...	See...
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Hit Pick Replication task	"Hit Pick Replication (Agilent Bravo)" on page 526
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Disassemble Vacuum (Agilent Bravo)

Description

The Disassemble Vacuum (Agilent Bravo) task (

) directs the robot to remove components from the Vacuum Filtration Station and place them back at the locations specified in the Assemble Vacuum task. If a filter plate was placed on top of the station, the Disassemble Vacuum task directs the robot to move the microplate to a specified location before disassembly.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

The task is used in conjunction with the following tasks:

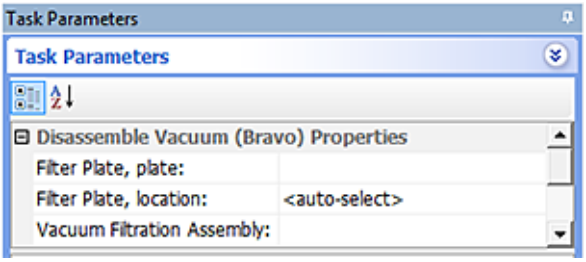
- Assemble Vacuum
- Move and Filter Plate
- Pipette Under Vacuum Collar
- Toggle Vacuum

Requirements

The Disassemble Vacuum task has the same requirements as the Assemble Vacuum task. The Assemble Vacuum task must precede the Disassemble Vacuum task in the protocol.

Task parameters

After adding the Disassemble Vacuum task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Filter Plate, plate	The labware used as the filter plate. Select the filter plate from the list, for example, the filter plate process.

Parameter	Description
Filter Plate, location	The deck location where you want the filter plate to be placed during the disassembly process. For example, if the Bravo profile is configured for a Filter Holder Plate accessory at deck location 4 or 6, you can specify the corresponding location. *Note:* If the station has two filter plates, one on top of the collar, and one immediately below the collar, make sure you specify the top filter plate. <auto-select> automatically places the labware at the first-available or appropriate location for the task. If you are performing double-filtration in configuration A, do not use auto-select.
Vacuum Filtration Assembly	The deck location of the Vacuum Filtration Station base.

Examples:

See the example in:

- "Move and Filter Plate (Agilent Bravo)" on page 568
- "Toggle Vacuum (Agilent Bravo)" on page 626

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • *Bravo Platform User Guide*
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Configuring the Vacuum Filtration Station	*Bravo Platform User Guide*
Setting the robot gripper offset in the Labware Editor	*VWorks Automation Control Setup Guide*
Move and Filter Plate	"Move and Filter Plate (Agilent Bravo)" on page 568
Toggle	"Toggle Vacuum (Agilent Bravo)" on page 626
Pipetting in a filter plate that sits under the collar	"Pipette Under Collar Begin and End (Agilent Bravo)" on page 584
Assemble Vacuum (Bravo) task	"Assemble Vacuum (Agilent Bravo)" on page 499
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343

For information about...	See...
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Dispense (Agilent Bravo)

Description

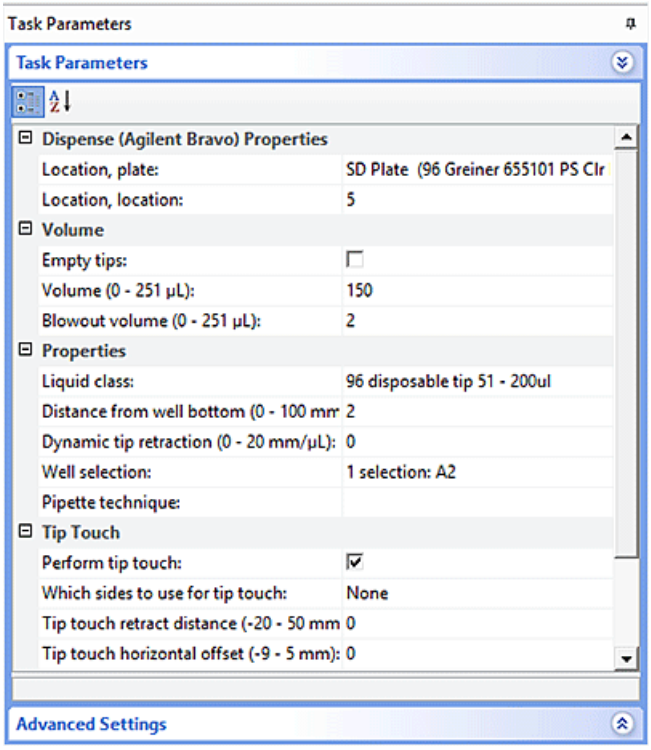
The Dispense (Agilent Bravo) (

) task dispenses liquid into a microplate, reservoir, or tubes in a tube rack.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Task parameters

After adding the Dispense task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Empty tips	The option to empty all liquid from the tips instead of using the dispense volume specification.
Volume (0–251 µL)	The volume of liquid to be dispensed from each pipette tip.
Blowout volume (0–251 µL)	Specifies the volume of air to dispense after the main volume has been dispensed while the tips are still in the wells. Typically, the blowout volume is the same as the pre-aspirate volume. Note: Blowout only occurs in the last quadrant dispensed for a given Dispense task.
Liquid class	The pipetting speed and accuracy. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Distance from well bottom (0–100 mm)	The distance between the end of the pipette tips and the well bottoms during the Dispense task. If you specify dynamic tip retraction, this is the starting distance. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the tips at the correct distance from the well bottom.
Dynamic tip retraction (0–20 mm/µL)	The rate at which to raise the pipette head during the Dispense task. Use dynamic tip retraction to prevent spills as the pipette tips displace the liquid. To move the tips: • *At the same rate as the volume change*. Calculate dynamic tip retraction (DTR) as follows: DTR = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTR > 1/A • *Slower than the volume change*. DTR < 1/A The starting and ending positions can be calculated as follows: $(V_{dispensed} * DTR) + Distance_{well\ bottom}$

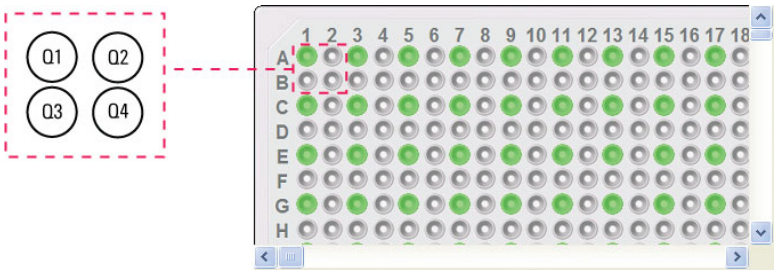
Parameter	Description
Pipette technique	The pipette location offset you want to use for the Dispense task. The list of pipette techniques are defined in the Pipette Technique Editor.
Perform tip touch	The option to touch the pipette tip on one or more sides of the well.
Which sides to use for tip touch	The side or sides of the well to use during tip touch: North, South, East, West, North/South, West/East, West/East/South/North.
Tip touch retract distance (–20 to 50 mm)	The vertical distance for the pipette tips to rise before touching the sides of the wells.
Tip touch horizontal offset (–9 to 5 mm)	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. For example, if you set a value of: • 0, the tips move a horizontal distance equal to the well radius • > 0, the tips attempt to move past the well radius, which results in a more forceful tip touch • < 0, the tips move a distance less than the radius of the well, resulting in a lighter tip touch

Quadrant pattern well selection

A quadrant is an evenly spaced array of locations that are accessible by the tips on a pipette head. The following table lists the types of pipette heads and the number of accessible quadrants in various microplates.

Pipette head channels/ pin tool pins	Microplate	Number of quadrants
96	96-well	1
	384-well	4
	1536-well	16
384	384-well	1
	1536-well	4
1536 (pin tool only)	1536-well	1

The following diagram demonstrates the concept of quadrants. The diagram shows a portion of a 384-well microplate and highlights the four quadrants (Q1, Q2, Q3, and Q4) that are accessible by the A1 tip of a 96-channel pipette head. Notice that the green color highlights all of the quadrant 1 (Q1) wells across the microplate.



Instead of a column- or row-wise pattern, you can select a quadrant pattern during well selection.

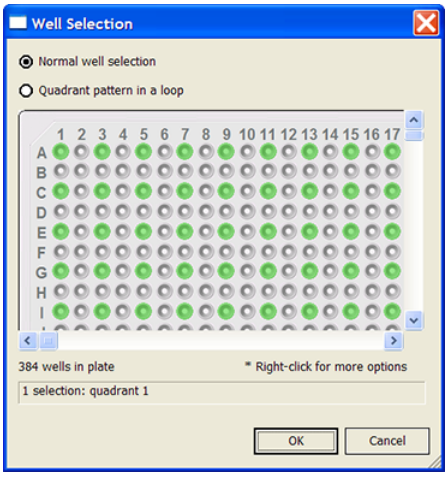
The quadrant pattern option is available only if:

- The number of channels in the pipette head (or pins in a pin tool) is fewer than the number of wells in the microplate. For example, you can use a 96-channel pipette head to dispense liquid into a 384-well microplate or 1536-well microplate.
- All the channels are selected in the Set Head Mode task when using a pipette head. (The Set Head Mode task is not an option when using a pin tool).
- The liquid-handling task is inside a loop.

IMPORTANT If you select a quadrant pattern, specifications in the Well Selection dialog box will override `task.Wellselection` values assigned in the Advanced Settings area.

To select a quadrant pattern:

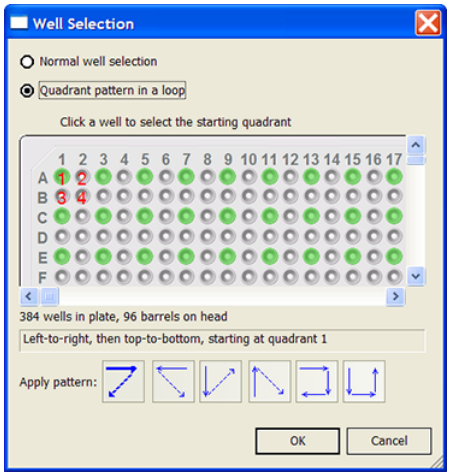
1 In the **Task Parameters** area, click the **Well selection** parameter box, and then click the Browse button. The Well Selection dialog box opens. By default, the **Normal well selection** option is selected. This option is used for column- and row-wise liquid-handling patterns.



2 Select **Quadrant pattern in a loop.** The contents of the dialog box change. Notice the following:

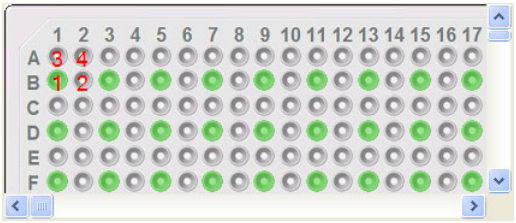
- Red numbers (1 through 4) appear on wells A1, A2, B1, and B2. The numbers indicate the pipetting sequence: 1 is the starting well, and 4 is the last well. In the following example, the sequence is A1, A2, B1, B2.
- Green wells indicate the starting well in the pipetting sequence.
- Pattern buttons at the bottom of the dialog box indicate the movement of the pipette channels. The movement description is provided in the text box above the buttons.

  Note: The last two patterns are unavailable if a group contains 16 wells. For example, the last two patterns are not available if you have a 96-well pipette head and a 1536-well microplate.



3 Select the starting well. The well becomes green and is labeled 1.

In the following example, the third quadrant (B1 well) is selected.



4 Click a pattern button to specify the pipette channel movement. After you click a pattern, the red numbers in the graphic are updated to show the sequence.

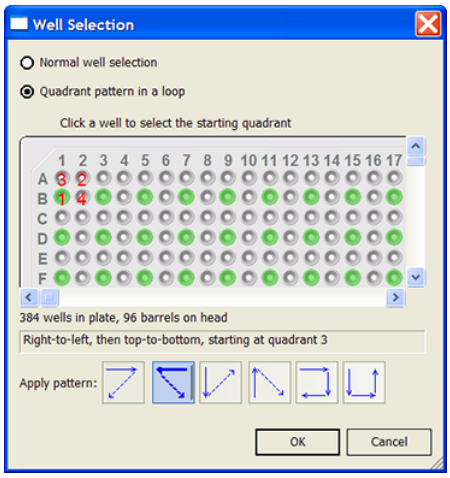
In the following example, the second pattern is selected (right-to-left, then top-to-bottom). The third quadrant (B1) is the starting well. The resulting movement is:

Quadrant 3 (B1)

Quadrant 2 (A2)

Quadrant 1 (A1)

Quadrant 4 (B2)



5 When you are finished, click **OK** to save the changes and return to the VWorks window.

Example: Dispense into a microplate on the Bravo Platform

Goal

Aspirate contents from a source microplate (Source 1) and dispense into a destination microplate.

Implementation

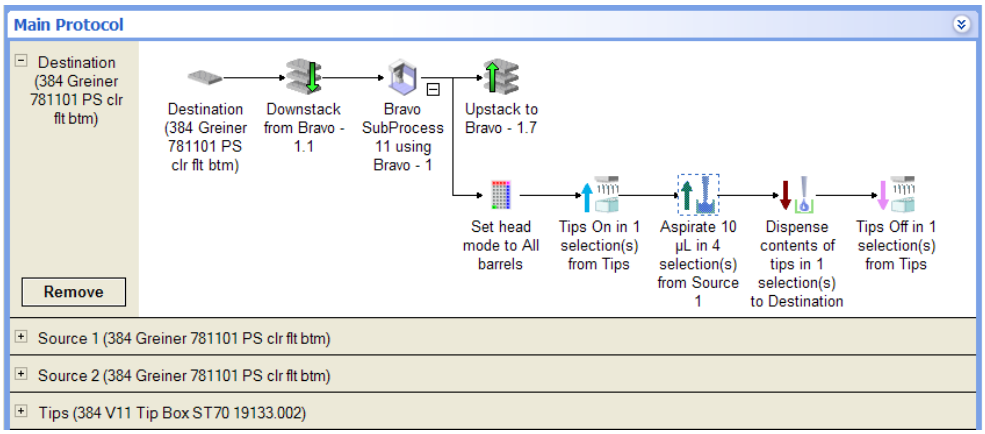
The Bravo deck is physically set up as follows:

- The destination microplates are at Bravo deck location 1.
- The source microplate is at deck location 2.
- The tip box is at deck location 9.

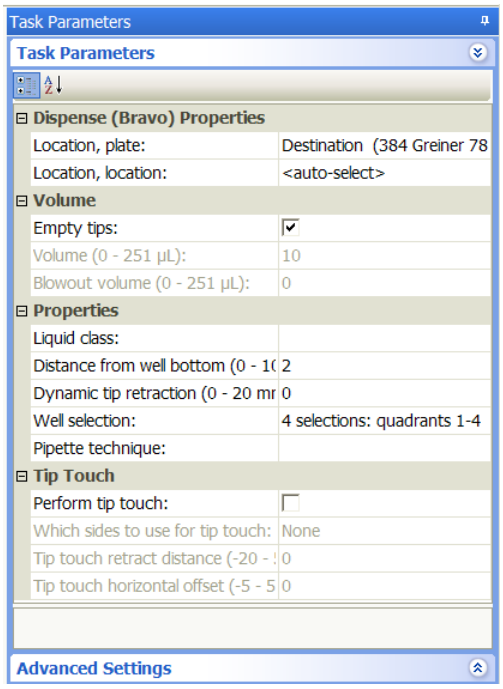
In the protocol, the following are added:

- Process for the destination microplate
- Configured labware for the source microplate
- Configured labware for the tip box

In the Destination plate process, a Bravo subprocess is added. Within the subprocess, an Aspirate task and a Dispense task are added as shown in the following example.



In the Dispense Task Parameters area, Destination is selected, because the goal is to dispense into that microplate.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • *Bravo Platform User Guide*
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Liquid classes	*VWorks Automation Control Setup Guide*
Pipette techniques	"Specifying pipetting techniques" on page 691
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Set Head mode task	"Set Head Mode (Agilent Bravo)" on page 601
Tips On task	"Tips On (Agilent Bravo)" on page 623
Tips Off task	"Tips Off (Agilent Bravo)" on page 619
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Dispense to Waste (Agilent Bravo)

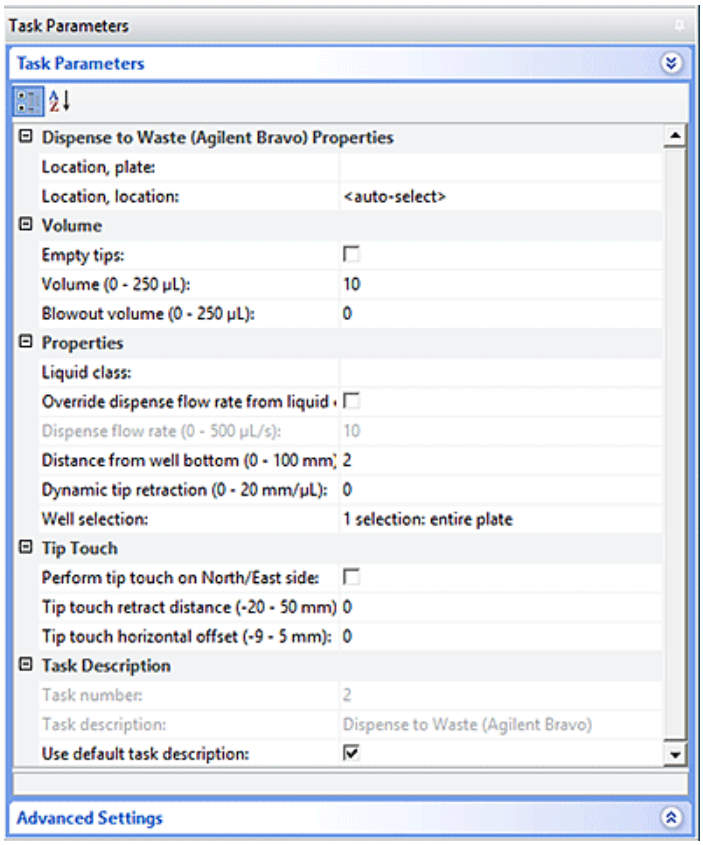
Description

The Dispense to Waste (Agilent Bravo) () task moves the pipettes by a horizontal offset and then dispenses used fluid in between the chimneys in the MicroWash Reservoir. The software calculates the horizontal offset automatically based on the labware definition for the MicroWash Reservoir (also known as the Tip Wash Station).

Task is available for...	Task is available in...
MicroWash Reservoir (Tip Wash Station) on Bravo Platform	Main Protocol, Bravo Subprocess

Task parameters

After adding the Dispense to Waste task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Empty tips	The option to empty all the liquid from the head into the wash station outside the chimneys.
Volume (0–250 µL)	The volume of liquid to be dispensed from each pipette.
Blowout volume (0–250 µL)	Specifies the volume of air to dispense after the main volume has been dispensed while the tips are still in the wash station.
Liquid class	The pipetting speed and accuracy. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Override dispense flow rate from liquid class	The option to override the dispense velocity specified in the liquid class. Selecting this option enables you to specify a value for the dispense flow rate without changing the liquid class.

Parameter	Description
Dispense flow rate (0–500 µL/s)	The numerical value or the JavaScript variable that will override the dispense velocity setting in the liquid class. A JavaScript variable enables the value to be assigned later, for example through a VWorks form. Using a VWorks form, an operator could easily change the flow rate for a dispense step in increments from as low as 1 µL/min up to 1001 µL/min using the same liquid class. If the task is included in a VWorks macro, a JavaScript variable enables you to change the value for the task at the macro level. **IMPORTANT** The software requires that the flow rate value be in microliters per second (µL/s) at run time. If you want an operator to enter the value in microliters per minute (µL/min), you can use scripting to convert the values for the software to use.
Distance from well bottom (mm)	The absolute distance between the end of the pipette tips and the bottom of the wash stationduring the Dispense to Waste task. If you specify dynamic tip retraction, this is the starting distance. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the tips at the correct distance from the well bottom.
Dynamic tip retraction (mm/µL)	The rate at which to raise the pipette head during the Dispense to Waste task. Use dynamic tip retraction to prevent spills as the pipette tips displace the liquid. To move the pipettes: • *At the same rate as the volume change*. Calculate dynamic tip retraction (DTR) as follows: DTR = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTR > 1/A • *Slower than the volume change*. DTR < 1/A The starting and ending positions can be calculated as follows: $(V_{dispensed} * DTR) + Distance_{well\ bottom}$

Parameter	Description
Perform tip touch on North/ East side	The option to touch the tip on an outer side of the adjacent northeast chimney or wall in the wash stationafter performing the dispense.
Tip touch retract distance (–20 to 50 mm)	The vertical distance for the pipette tips to rise before performing the tip touch.
Tip touch horizontal offset (–9 to 5 mm)	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. For example, if you set a value of: • 0, the tips move a horizontal distance equal to the well radius • > 0, the tips attempt to move past the well radius, which results in a more forceful tip touch • < 0, the tips move a distance less than the radius of the well, resulting in a lighter tip touch

Example: Dispense into the wash station using a Bravo 96AM Head

Goal

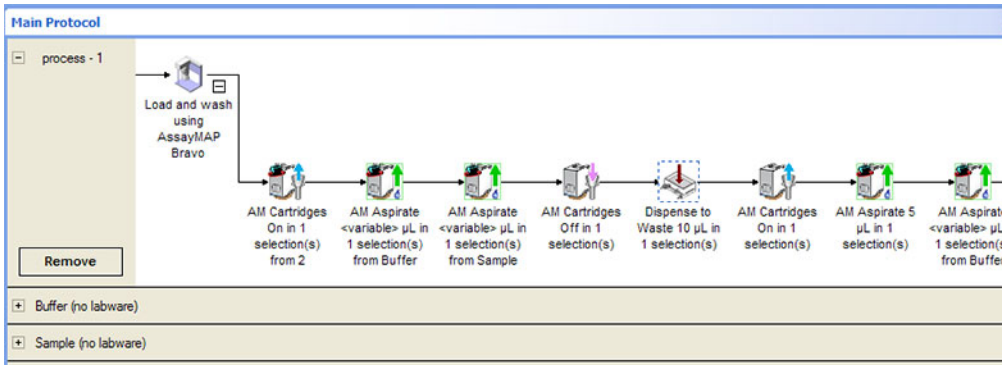
After aspirating fluids up through mounted AssayMAP Bravo cartridges, remove the cartridges, dispense the contents of the Bravo 96AM Head syringes at the wash station, remount the cartridges for additional aspiration steps.

Implementation

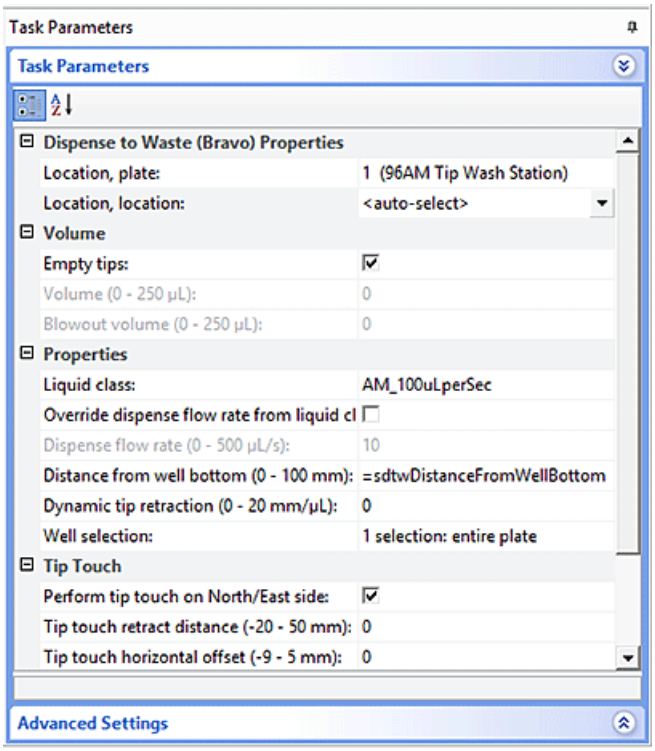
In Bravo Diagnostics, set up the autofilling station (wash station) on a deck location. Associate the station with one or more pump modules.

In the protocol, add a configured labware for the wash station in addition to the other processes and configured labware required by the protocol. In the example shown, the configured reservoir is called Wash.

In the Bravo subprocess where the liquid-handling tasks are specified, a Dispense to Waste task is added after a series of aspiration tasks.



In the Dispense to Waste Task Parameters area, the wash station location is selected so that the task is performed in this labware.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • *Bravo Platform User Guide*
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Liquid classes	*VWorks Automation Control Setup Guide*
Pipette techniques	"Specifying pipetting techniques" on page 691
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Set Head mode task	"Set Head Mode (Agilent Bravo)" on page 601
Tips On task	"Tips On (Agilent Bravo)" on page 623
Tips Off task	"Tips Off (Agilent Bravo)" on page 619
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Hit Pick Replication (Agilent Bravo)

Description

The Hit Pick Replication (Agilent Bravo) ( Hit Pick Replication (Agilent Bravo)) task allows you to:

- Hit pick, or transfer contents, from selected wells in a source microplate to a destination microplate.
- Replicate a microplate by transferring contents from a source microplate into multiple destination microplates.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Workflow

The following table presents the steps for using the Hit Pick Replication task.

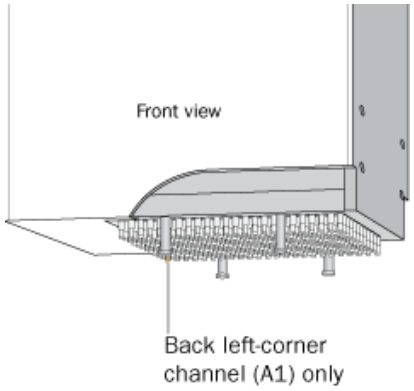
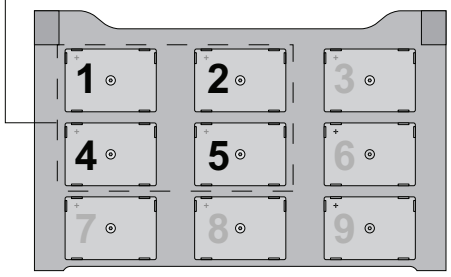
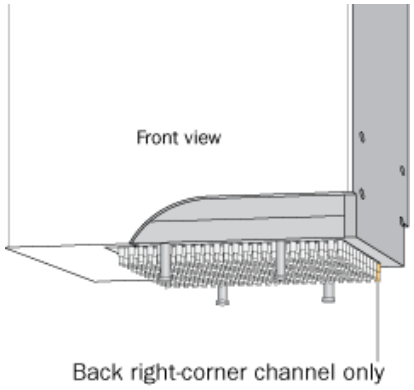
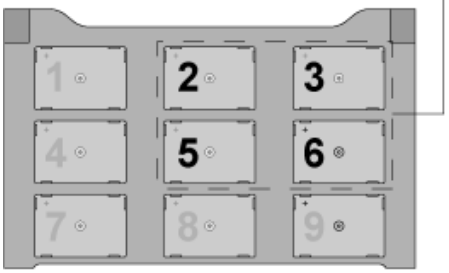
Step	For this task...	See...
1	Set up labware on the Bravo deck.	"Setting up labware on the Bravo deck" on page 527
2	Create an input file.	"Creating input files" on page 528
3	*Optional*. Import the input file. The input file can be stored in the local file system or imported into Shared Services storage. You specify the input file location in Protocol Options.	"Importing and exporting miscellaneous files" on page 49
4	Create the protocol.	"Creating a protocol: basic procedure" on page 17
5	Add the Hit Pick Replication task, and select or create a format file.	"Adding the Hit Pick Replication task" on page 530 "Creating format files" on page 540
6	Verify source-destination transfers.	"Verifying source-destination transfers" on page 556

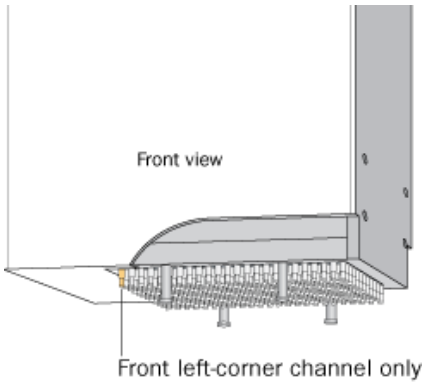
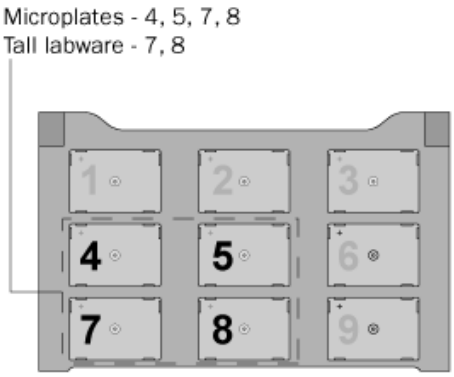
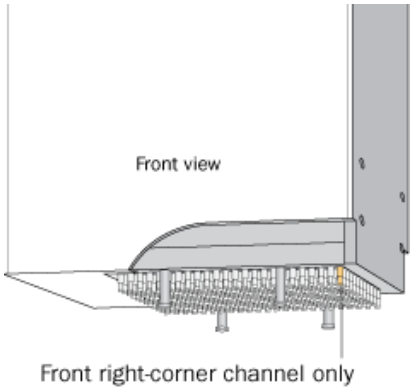
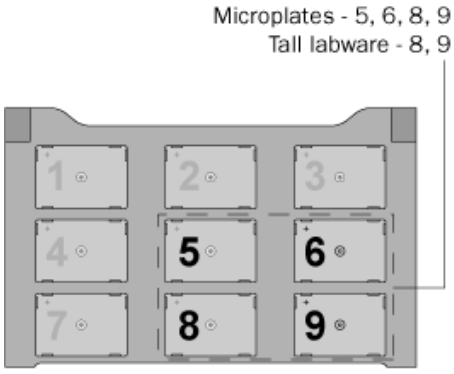
Setting up labware on the Bravo deck

For hit-picking transfers, the deck locations you can access depends on the pipette head channel you select. In addition, because tip boxes and reservoirs are tall, you must consider the locations of the these labware relative to the source and destination microplates.

The following table shows the channel selection and corresponding deck location limits for both microplates and tall labware. Use this information when setting up labware for the hit-picking task.

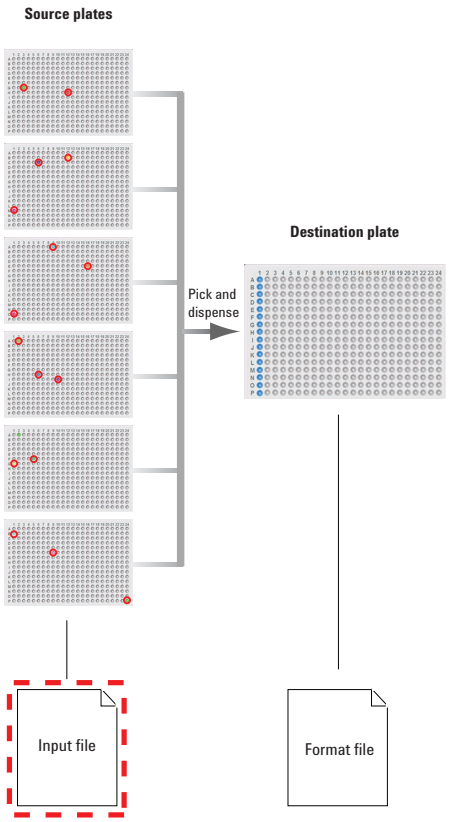
For example, if you select the A1 channel, you can place microplates at locations 1, 2, 4, and 5. If you have tall labware, such as a tip box and a reservoir, place them at locations 1 and 2. Doing so permits the pipette head to access the microplates at locations 4 and 5 without colliding with the tall labware.

Pipette channel selection	Accessible deck locations
	Microplates - 1, 2, 4, 5 Tall labware - 1, 2 
	Microplates - 2, 3, 5, 6 Tall labware - 2, 3 

Pipette channel selection	Accessible deck locations
	
	

Creating input files

Both the hit-picking and microplate replication tasks require an input file. An input file contains information about the source microplates.



You must have the required input file before adding the Hit Pick Replication task or before you start the protocol run. The input file can be from a LIMS or created manually. The file must meet the following requirements:

- The file must be in the comma-separated value (CSV) format.
- The file can contain a heading in the first row, or the first few consecutive rows.
- *Microplate replication*. The input file must contain a column that lists source microplate barcodes or IDs.
- *Hit picking*. The input file must contain the following columns, in any order:
  - Source microplate barcode or ID
  - Well locations (in a single column or across two columns)
  - Variable dilution factors (if dilution series will be created)
  - Transfer-volume information (if variable-volume transfer is desired)

IMPORTANT

Hits from each source microplate must be in consecutive rows in the input file. In addition, the order in which the microplates are processed in protocol runs must match the order of the microplates in the input files.

The following examples show two input files (1 and 2) displayed in Microsoft Excel. The heading is in the first row in both files. Microplate IDs are used instead of barcodes. Notice that the well locations are displayed differently in the two files. The file on the right (2) also contains dilution factor information.

1

	A	B	C
1	PlateID	Well	Volume
2	Plate 1	H12	20
3	Plate 1	G3	30
4	Plate 2	C6	20
5	Plate 2	B12	15
6	Plate 2	M1	20
7	Plate 3	A9	20
8	Plate 3	E16	30
9	Plate 3	O1	15
10	Plate 4	H6	20
11	Plate 4	I10	30
12	Plate 5	A2	20
13	Plate 5	F5	20
14	Plate 5	G1	20
15	Plate 6	F9	15
16	Plate 6	B1	30
17	Plate 6	P24	20

2

	A	B	C	D
1	PlateID	Well row	Well col	Dil factor
2	Plate 1	H	12	2
3	Plate 1	G	3	2
4	Plate 2	C	6	3
5	Plate 2	B	12	4
6	Plate 2	M	1	3
7	Plate 3	A	9	1
8	Plate 3	E	16	2
9	Plate 3	O	1	3
10	Plate 4	H	6	4
11	Plate 4	I	10	2
12	Plate 5	A	2	4
13	Plate 5	F	5	3
14	Plate 5	G	1	2
15	Plate 6	F	9	3
16	Plate 6	B	1	1
17	Plate 6	P	24	4

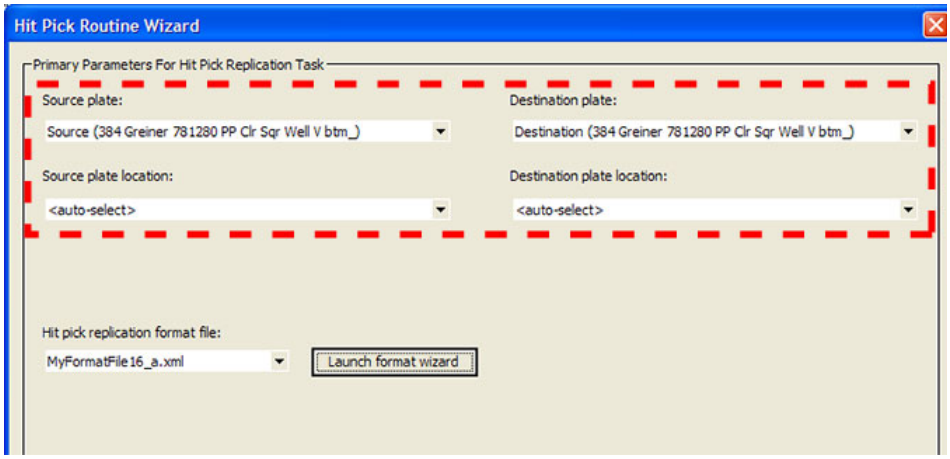
The input file can be stored in the local file system or you can import the file into Shared Services storage. To import the input file, see "Importing and exporting miscellaneous files" on page 49. To use Shared Services storage, ensure that you select the Use Shared Services storage property in Protocol Options. For details, see "Setting protocol options" on page 42.

Adding the Hit Pick Replication task

When you add the Hit Pick Replication () task, the software automatically starts the Hit Pick Routine Wizard.

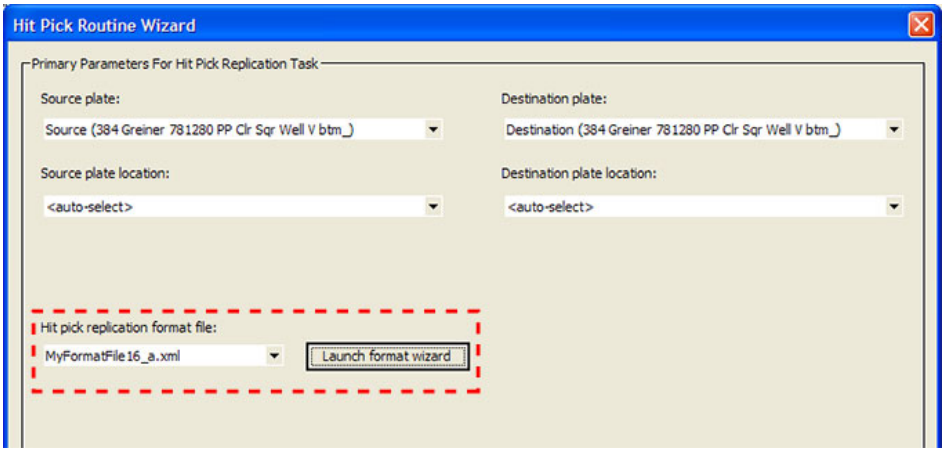
In the Hit Pick Routine Wizard:

1 Select the source and destination microplates and their locations.



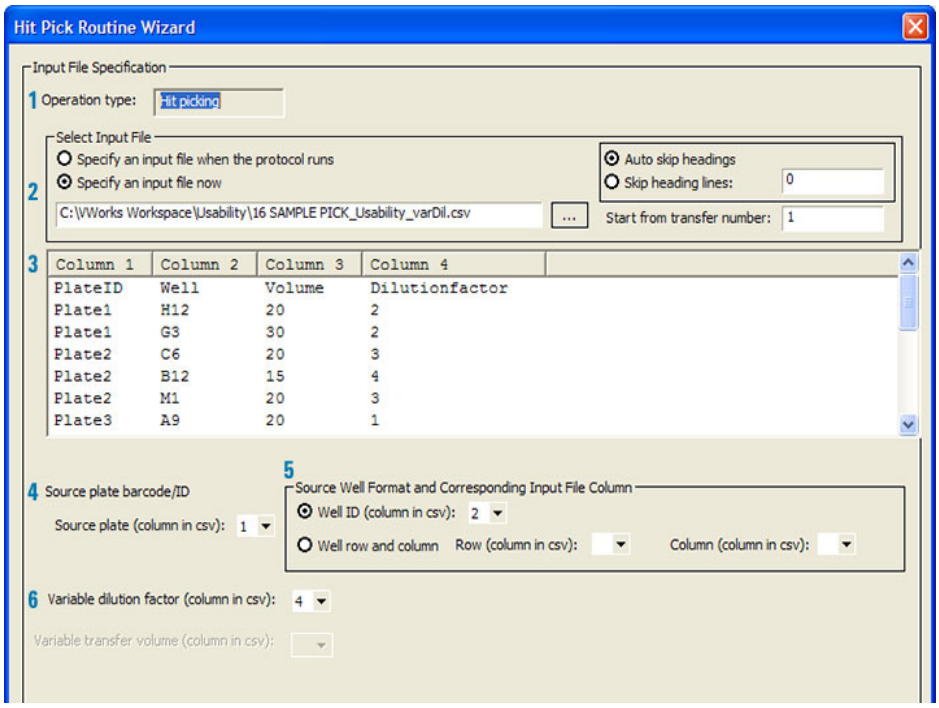
2 Do one of the following:

- Select an existing format file from the **Hit pick replication format** list.
- Click **Launch format wizard** to create a new format file. See "Creating format files" on page 540 for detailed instructions.



When you are finished, click **Next.**

3 Supply the input file information. If you do not have an input file, see "Creating input files" on page 528.

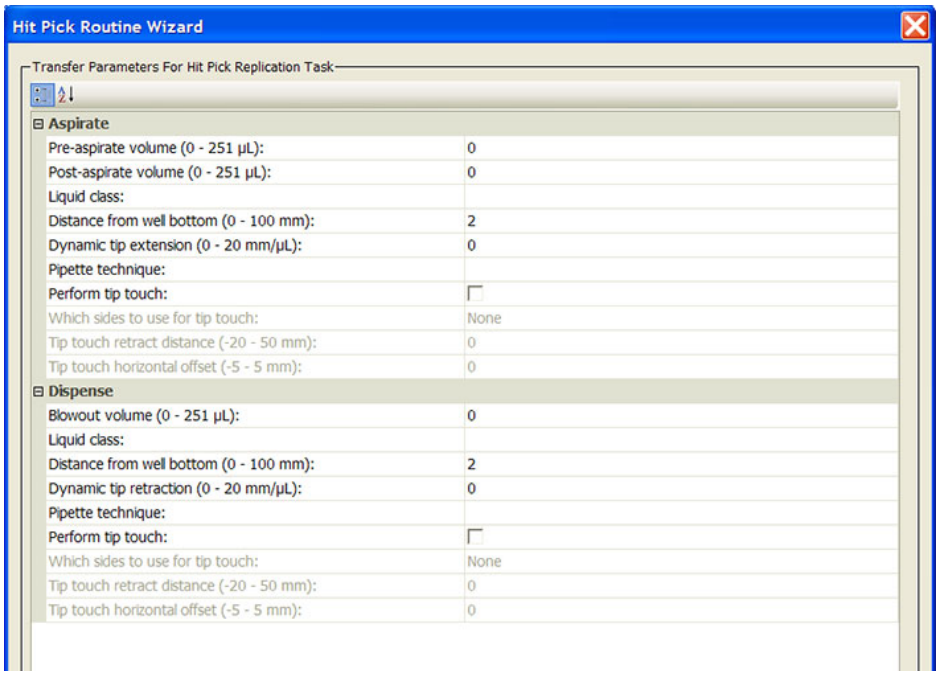


Item	Input file specification	Description
1	Operation type	Displays the type of operation: Hit picking, or replication. The selection is made in the format file and cannot be changed on this page. For information about the format file, see "Creating format files" on page 540.

Item	Input file specification	Description
2	Select input file	Allows you to select one of the following: • **Specify an input file when the protocol runs** • **Specify an input file now** If you select **Specify an input file now**, use the browse button to locate and select the desired input file. The list of available files includes only the files (.csv) in the local file system or only files in Shared Services storage depending on how you set the Use Shared Services storage property in Protocol Options. For details, see "Setting protocol options" on page 42.
		In either case, you can select one of the following: • **Auto skip heading**. Enables the software to automatically find and skip rows that contain heading information. • **Skip heading lines**. Skips the specified number of rows at the top of the input file. Select this option if you want to manually specify the number of heading rows.
		In addition, you can specify the row to start the transfer from in the **Start from transfer number** box. 1 indicates starting the first transfer from the first row after the heading rows.
3	Input file contents	Displays the contents of the input file.
4	Source plate barcode/ID	Allows you to select the column that contains the source plate barcode or ID.
5	Source well format and corresponding input file column	*Hit-picking only*. Allows you to indicate whether the well location is specified as: • *Well ID*. The well location is in a single column (for example, A24). Select the column that contains the well ID. • *Well row and column*. The well location is in two columns. Select the columns that contain the well row and well column information. The row information can be a letter or a number.
6	Variable dilution factor and Variable transfer volume	*Hit-picking only*. Allows you to select the column that contains the dilution factors and transfer volume.

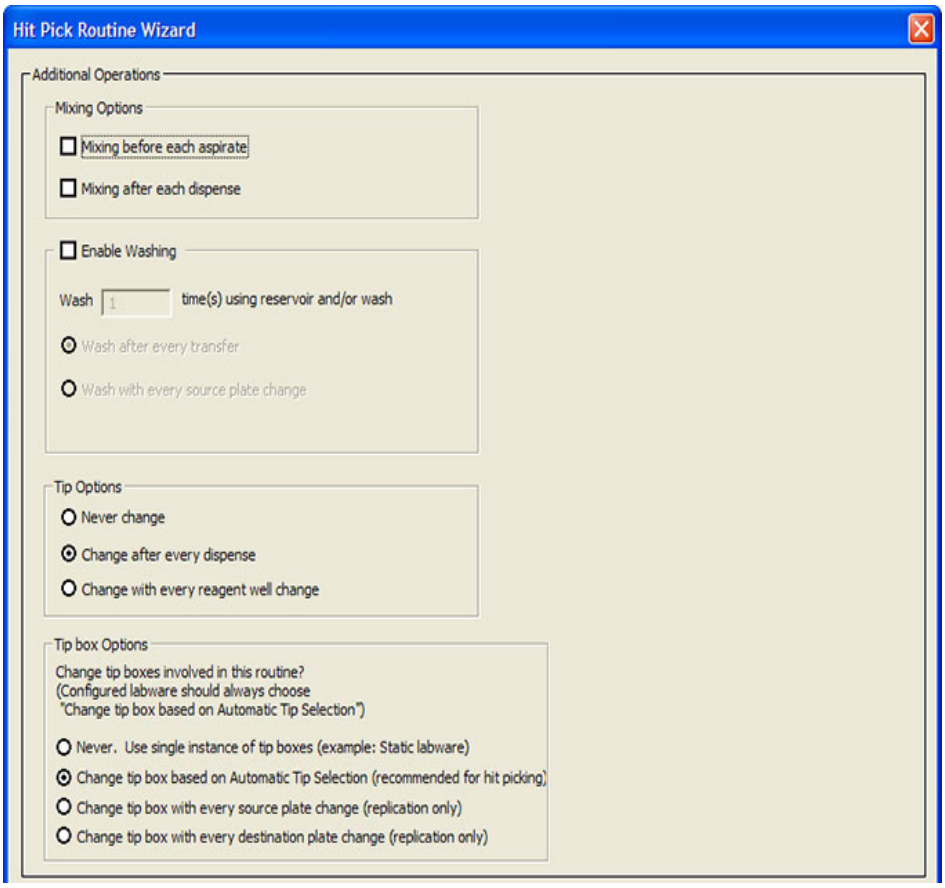
When you are finished, click **Next.**

7 Specify the Aspirate and Dispense task parameters. For detailed parameter descriptions, see "Aspirate (Agilent Bravo)" on page 455 and "Dispense (Agilent Bravo)" on page 514.

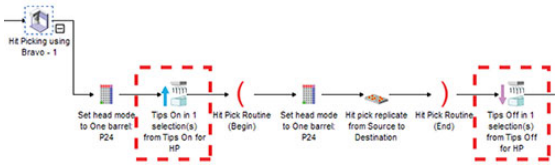


When you are finished, click **Next.**

8 Select additional operations, if applicable.

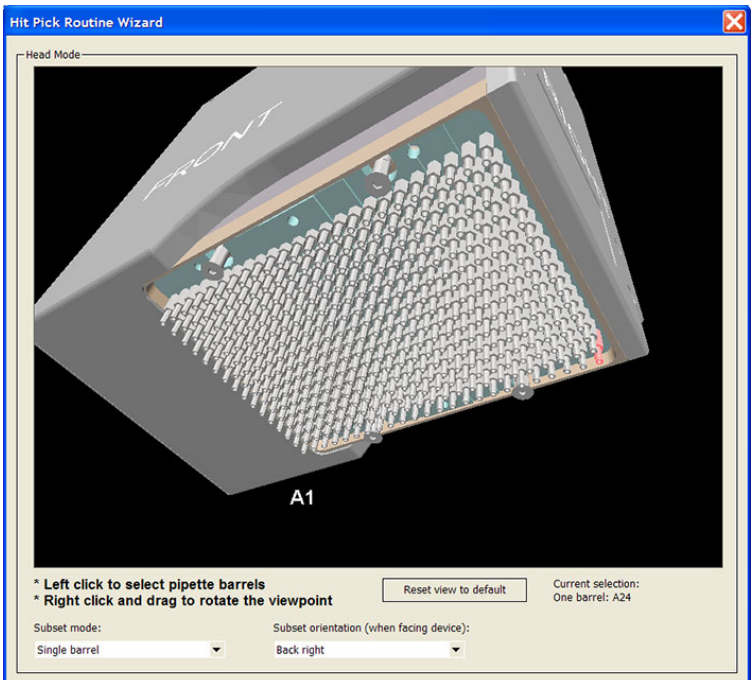


Additional operation	Description
Mixing Options	Select where you want to add a Mix task: • Before each aspirate task • After each dispense task You will need to set the Mix parameters later in the wizard. If you do not want to mix, clear both check boxes.
Enable Washing	Select the check box if you want to add one or more Wash Tips tasks. You can specify the number of tip washes and whether you want to: • Wash after every transfer • Wash with every source microplate change *Note:* This option only works with an autofilling wash station, for example, a microwash reservoir or tip wash station. You will need to set the Wash Tips parameters later in the wizard. Clear the check box if you do not want to add any Wash Tips tasks.

Additional operation	Description
Tip Options	Select the tip-change option: • *Never change*. Does not change the tips within the hit-picking routine. *Note:* If you select Never change, you must add the Set Head Mode, Tips On, and Tips Off tasks in the subprocess, outside of the hit-picking routine, after you are finished with the wizard.  • *Change after every dispense*. Changes the tips after every dispense within the hit-picking routine. • *Change with every source well change*. Changes the tips after every source well change during the hit-picking routine.
Tip Box Options	Select when to change the tip box: • *Never. Use single instance of tip boxes (example: Static labware)*. Does not change the tip box within the hit-picking routine. • *Change tip box based on Automatic Tip Selection (recommended for hit picking)*. Uses tip-tracking within the hit-picking routine, and changes the tip box when the box is fully used. Select this option if the tip boxes you are using for hit-picking are configured labware. *Note:* Tip-tracking continues in the Dilute to Final Volume routine. • *Change tip box with every source plate change (replication only)*. Changes the tip box every time the source microplate changes. This option is available only when a replication format file is selected. • *Change tip box with every destination plate change (replication only)*. Changes the tip box every time the destination microplate changes. This option is available only when a replication format file is selected.

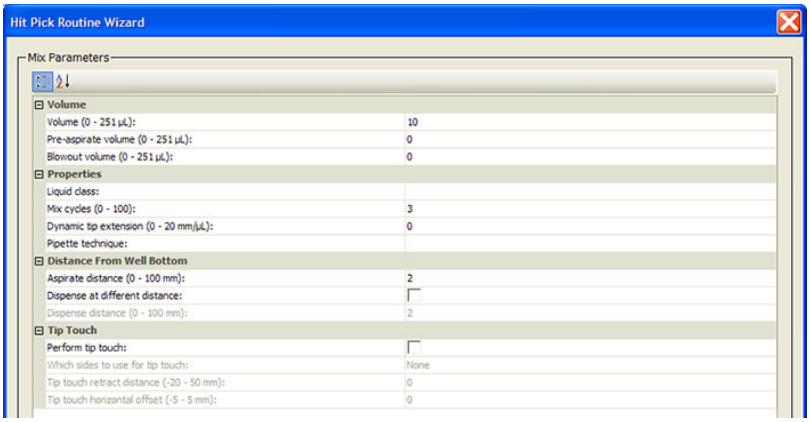
When you are finished, click **Next.**

9 Select the pipette channels to use. For a description of the selections, see "Set Head Mode (Agilent Bravo)" on page 601.



When you are finished, click **Next.**

10 *If you selected mixing options in step 8*. Set the mixing parameters. For a description of the parameters, see "Mix (Agilent Bravo)" on page 559.



When you are finished, click **Next.**

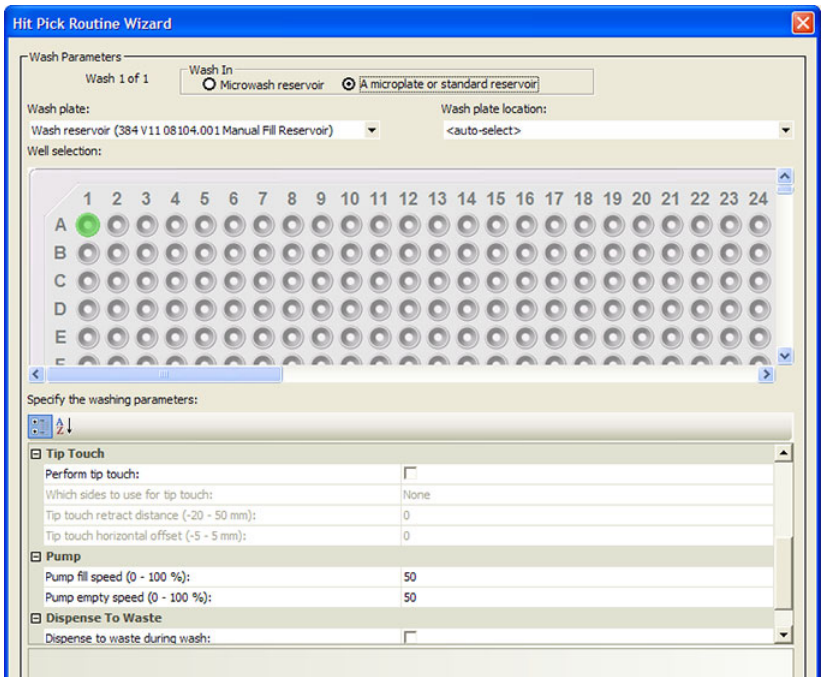
11 *If you selected wash options in step 8*. Set the wash parameters.

a In the **Wash In** area, select the type of labware to be used for the wash task:

- *MicroWash Reservoir (wash station)*. Adds Wash Tips tasks and will use either configured or static labware.
- *A microplate or standard reservoir*. Adds Mix tasks.

b In the **Wash plate** list, select the desired labware and location.

c Set wash-tip parameters. For a description of the parameters, see "Wash Tips (Agilent Bravo)" on page 632.



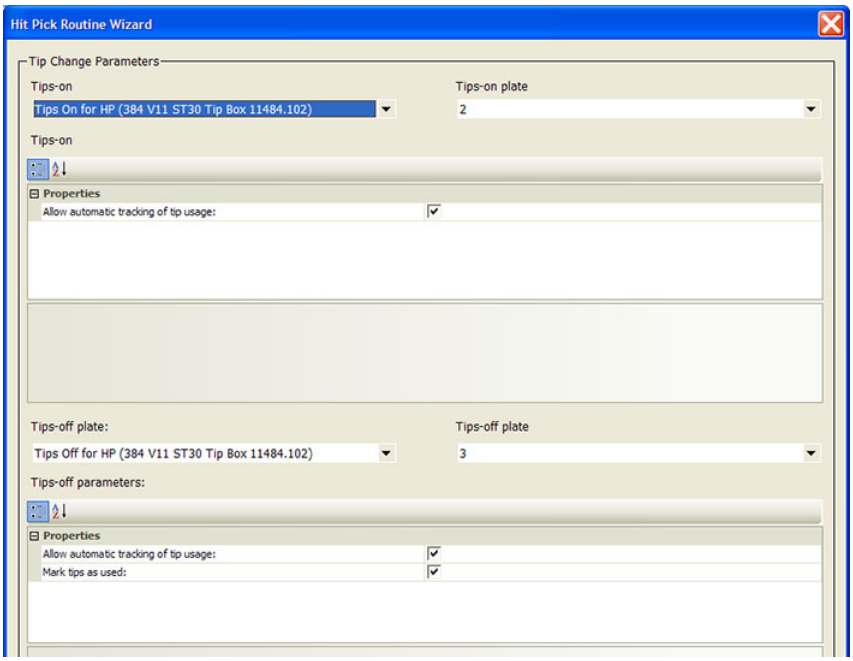
When you are finished, click **Next.**

12 *If you selected a tip-change option in step 8*. Select the tip box options.

a Select the tip boxes you want to use.

b Select the tip-tracking options you want to use:

- *Allow automatic tracking of tip usage*. Allows the software to track pipette tip usage during the protocol run or across different protocol runs. If you select the option in this task, you must also select the option in the Tips Off task. In general, use the default selections displayed.
- *Mark tips as used*. The option to use only new pipette tips during the protocol run. Select the option so that the software counts the number of tips used during the protocol run. The tips that have been used once are marked as used so that they cannot be reused. Clear the check box so that during the next Tips On task, the same tips can be reused. In general, use the default selections displayed.



13 Click **Finish.**

Based on the information you provide in the wizard, the software adds the tasks that are necessary to produce the desired hit-picking or replication results.

Notice the following:

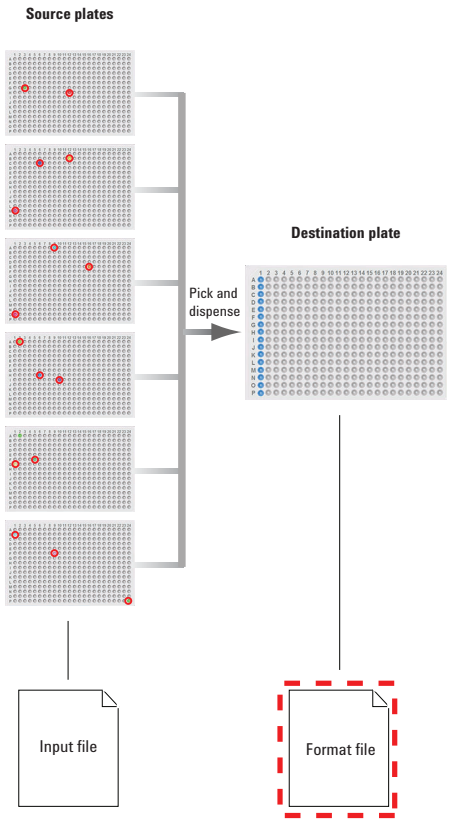
- The Hit Pick Routine (Begin) and Hit Pick Routine (End) tasks mark the beginning and end of the microplate-replication or hit-picking tasks.
- You can modify any of the task parameters directly in the Task Parameters area.
- You can add more tasks to or delete tasks from the routine.
- You can return to the Hit Pick Routine Wizard. To do this, select either the Hit Pick Routine (Begin) or Hit Pick Routine (End) task. In the Task Parameters area, click **Launch hit pick routine wizard.**

CAUTION **Whenever you return to the Hit Pick Routine Wizard, make modifications, and finish the wizard, a new set of tasks will replace the existing tasks in the routine.**

Creating format files

About format files

For both microplate replication and hit picking, you are required to use a format file. The format file shows the dispense pattern in destination microplates.



Workflow

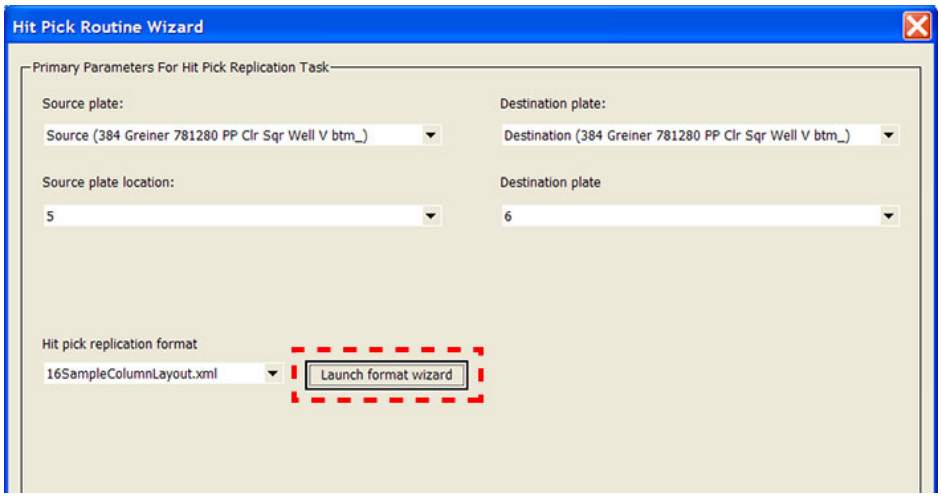
You use the Hit Pick Replication Format Wizard to create the format file. You can access the wizard when you add the Hit Pick Replication task. The overall workflow for creating the format file is:

Step	For this task...	See...
1	Create the format file.	"Creating format files" on page 540
2	Add contents to the format file for one of the following: • Hit picking, and dilution series (if desired) • Replication	One of the following: • "Adding hit-picking information in the format file" on page 541 • "Adding replication information in the format file" on page 548

Creating new format files

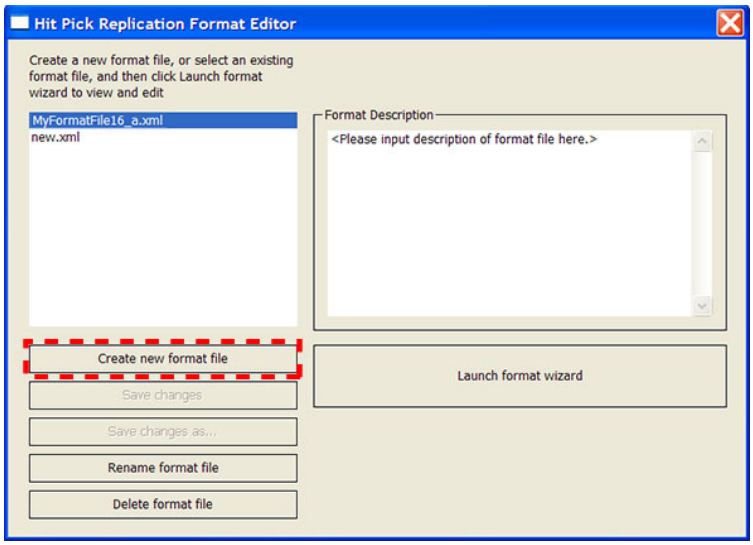
To create a new format file:

1 On the first page of the Hit Pick Routine Wizard, click **Launch format wizard.**



The Hit Pick Replication Format Editor opens.

2 Click **Create new format file,** type a name for the file in the prompt dialog box, and then click **OK.** The new format file name appears in the list.



The software saves the format file in XML format in a folder path that is set when the software is installed, for example:

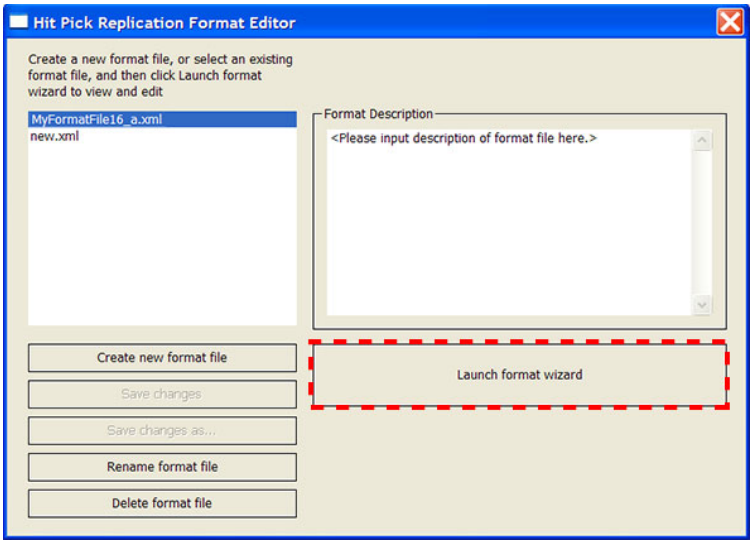
/VWorks Projects/VWorks/Pipetting/Hit Pick Format Files/

The file path of the storage location is displayed in the Options dialog box. For details, see "Viewing and setting file directory paths" on page 299.

Adding hit-picking information in the format file

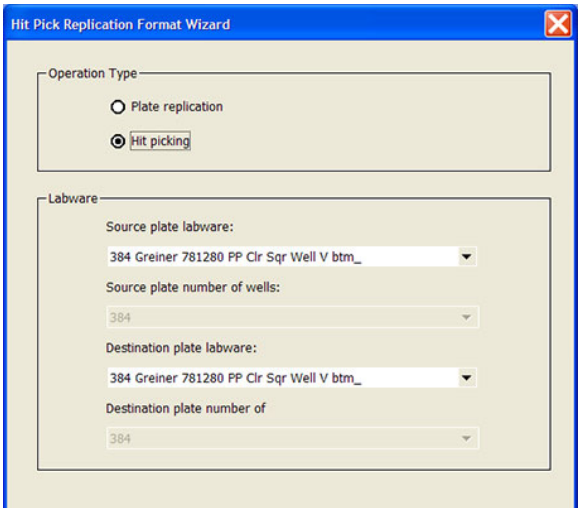
To add hit picking information in the format file:

1 In the **Hit Pick Replication Format Editor,** make sure the format file name is selected, and then click **Launch format wizard.**



The Hit Pick Replication Format Wizard opens.

2 On the first page of the wizard:

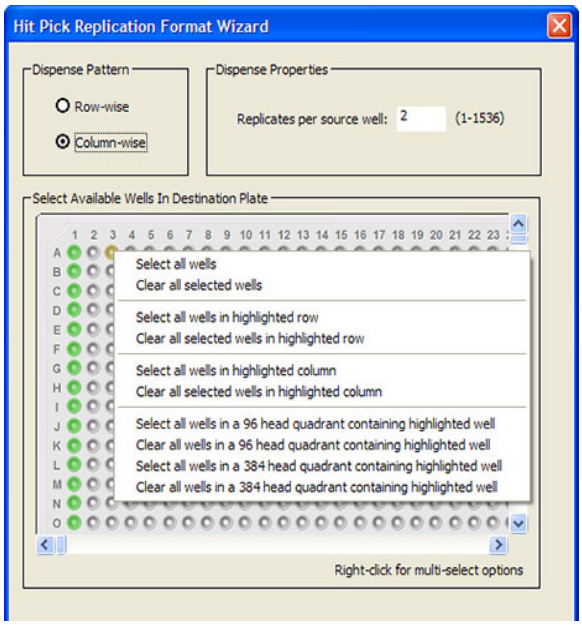


a In the **Operation Type** area, select **Hit picking.**

b In the **Labware** area, select the **Source plate labware** and the **Destination plate labware.**

c Click **Next.**

3 On the dispense information page:



a In the **Dispense Pattern** area, select one of the following dispense methods:

- Row-wise
- Column-wise

b In the **Dispense Properties** area, type the number of **Replicates per source well,** if applicable.

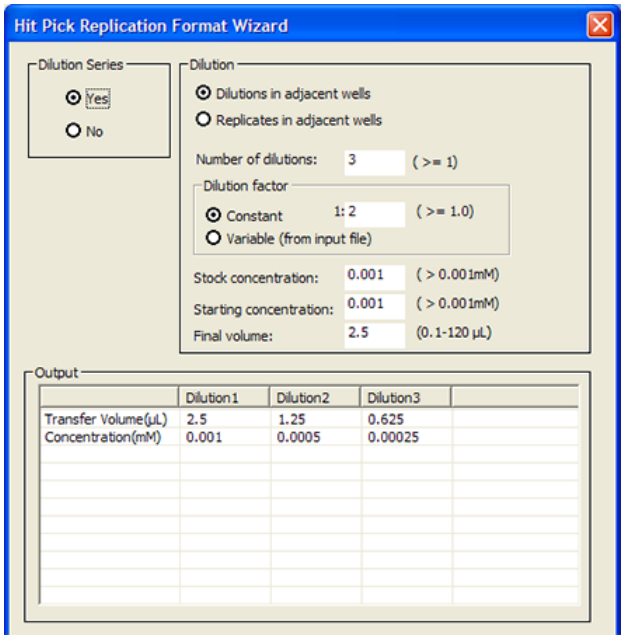
c In the destination plate graphic, select the wells into which the source contents will be dispensed. Click a well to individually select or clear the well. Alternatively, right-click a desired well and select a command from the menu that appears.

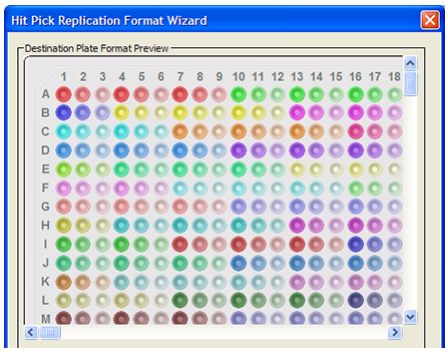
d Click **Next.**

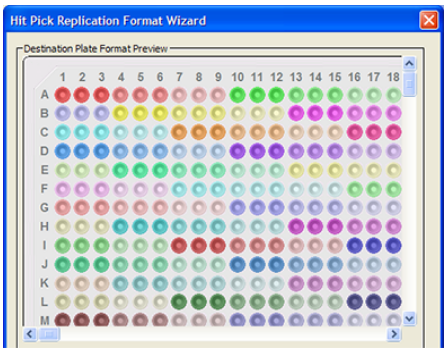
4 On the dilution series page:

a In the **Dilution Series** area, select one of the following:

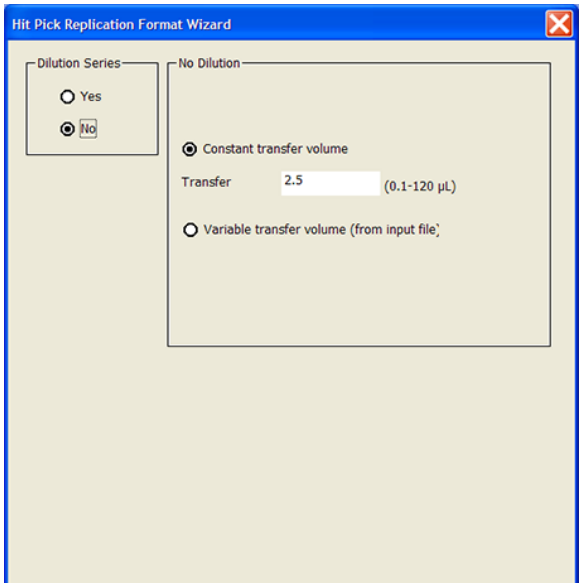
- *Yes*. If you select **Yes,** you must supply additional information about the dilution series.



Dilution selection or parameter	Description
Dilution in adjacent wells	Dispenses the dilution in adjacent wells before replicating the same series, as the following example shows. Notice that the dispense pattern is row-wise. 

Dilution selection or parameter	Description
Replications in adjacent wells	Dispenses the replicates in adjacent wells before continuing the dilution series, as the following example shows. Notice that the dispense pattern is row-wise. 
Number of dilutions	Specifies the number of dilutions to perform. A value of 1 means no dilution is performed.
Dilution factor	Uses the specified dilution factor: • *Constant*. The same dilution factor is used throughout the process. Type the factor to use in the box. For example, a factor of 1:2 dilutes the starting concentration by one-half. If the number of dilutions is 3, then the concentrations in the series are 1, 1:2, and 1:4. • *Variable (from input file)*. Different factors are used during the process. The factors are specified in the input file.
Stock concentration	Specifies the concentration of the stock solution.
Starting concentration	Specifies the concentration of the starting concentration.
Final volume	Specifies the concentration of the final volume in each well. The software will use this value to calculate the volume for prefilling or backfilling. See "Dilute to Final Volume (Agilent Bravo)" on page 504.

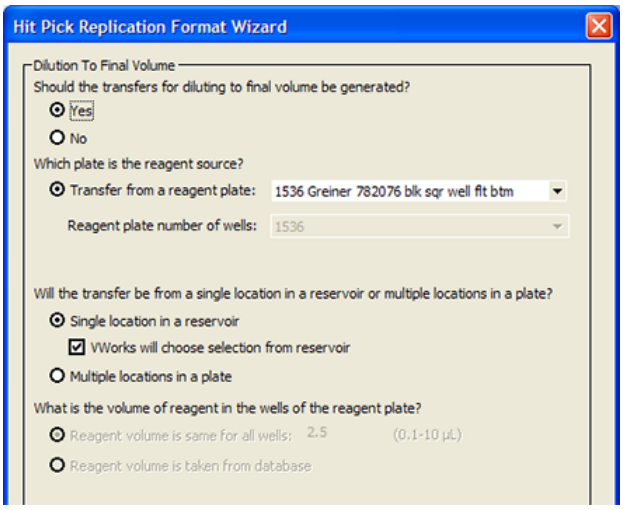
– *No*. If you select **No,** you must select the type of volume transfer.



Dilution selection or parameter	Description
Constant transfer volume	Uses the same transfer volume throughout the process.
Variable transfer volume	Uses different transfer volumes during the process. The volumes are specified in the input file.

b Click **Next.**

5 *If you selected dilution series in step 4*. Select the desired options for dilute to final volume.



a Select whether you want to backfill or prefill the destination microplate. If you select **Yes**, you must supply additional information. If you select **No**, you can proceed to step d.

b Select the reagent microplate to use for the backfilling or prefilling process.

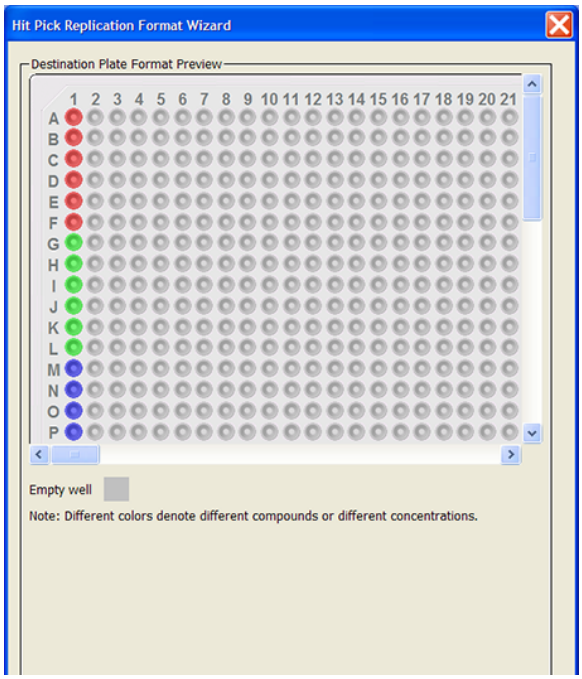
c Select whether the transfer will be from one of the following:

- **Single location in a reservoir.** Some reservoir designs might limit the pipette's access. You can allow the software to determine how to access the reservoir, based on the labware definition. To do this, select **VWorks will choose selection from reservoir.** Alternatively, you can clear the check box and manually select the access point in the reservoir, using the microplate graphic displayed. The wells in the microplate represent different access points in the reservoir.
- **Multiple locations in a plate.** If the transfer will be from a microplate, you can select the wells from which to aspirate. In addition, you can specify a constant reagent volume, or indicate that the reagent volume information will be obtained from a database.

d Click **Next.**

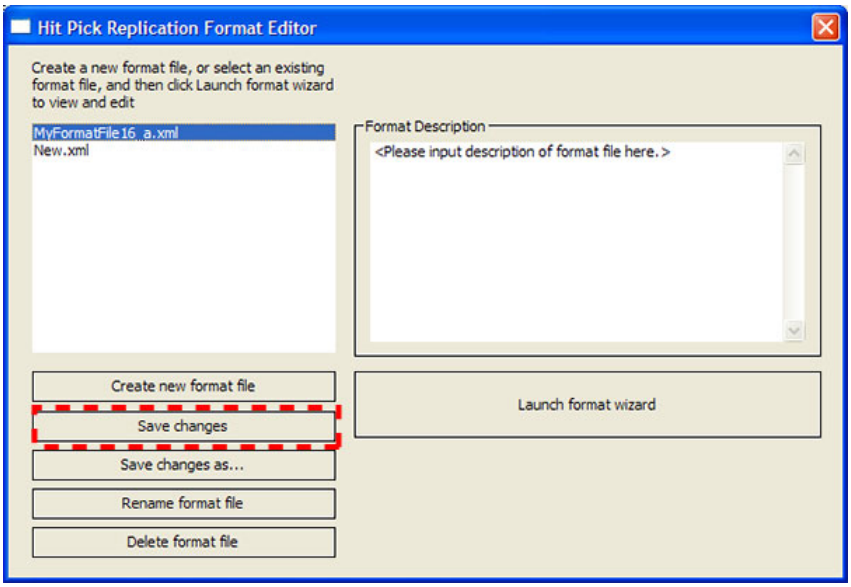
6 On the destination plate preview page, confirm that the dispense pattern is correct.

Note: The dispense pattern shown on the screen is a preview of the *first* destination microplate only. To check the mapping in all of the destination microplates, compile the protocol, and then view the output files. Alternatively, you can check the Pipette Log after running the protocol in simulation mode. For more information about output files and the Pipette Log, see "Verifying source-destination transfers" on page 556.



When you are done, click **Finish**.

7 Back in the **Hit Pick Replication Format Editor,** make sure the correct format file name is selected, and then click **Save changes.** If you altered an existing format file, you can click **Save changes as** to save it as a different file.

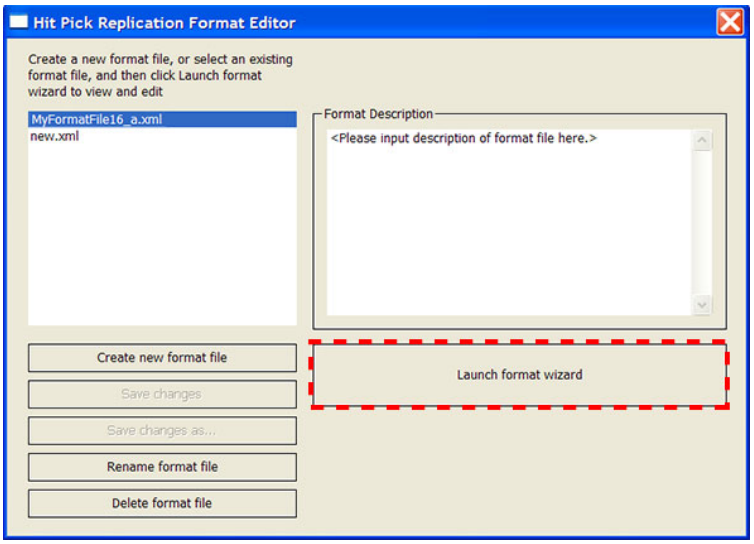


8 Click the close button ( ) at the top right corner of the dialog box to return to the Hit Pick Routine Wizard. To continue in the Hit Pick Routine Wizard, return to step 3 in "Adding the Hit Pick Replication task" on page 530.

Adding replication information in the format file

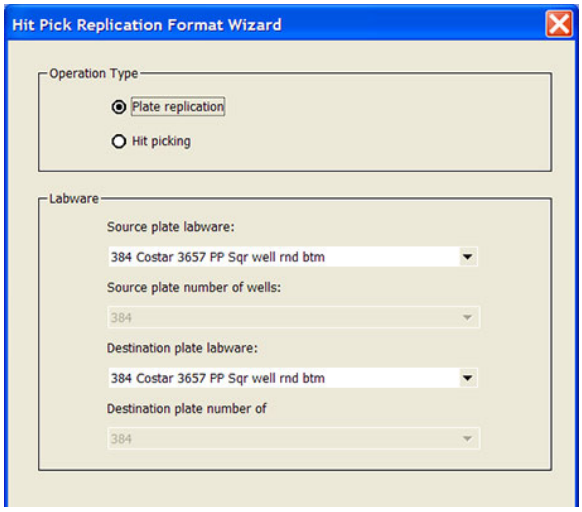
To add replication information in the format file:

1 In the **Hit Pick Replication Format Editor,** make sure the format file name is selected, and then click **Launch format wizard.**



The Hit Pick Replication Format Wizard opens.

2 On the first page of the wizard:

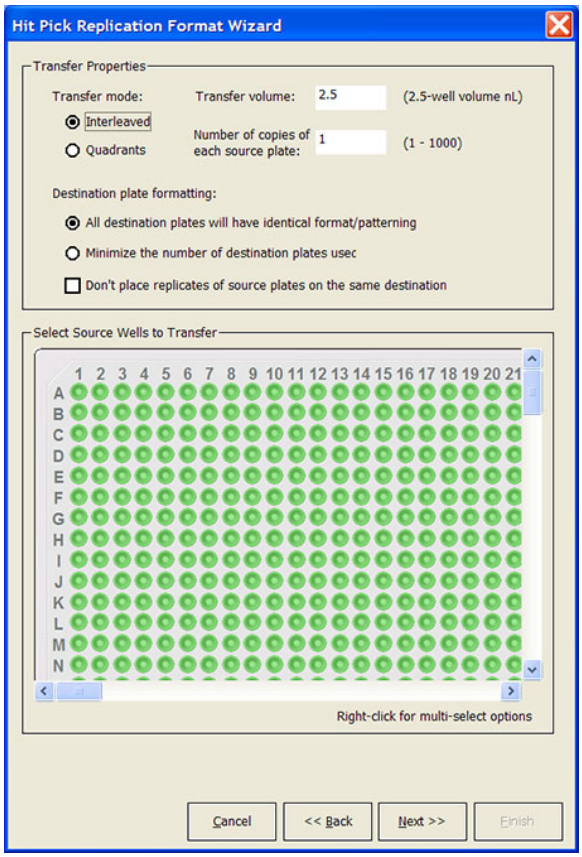


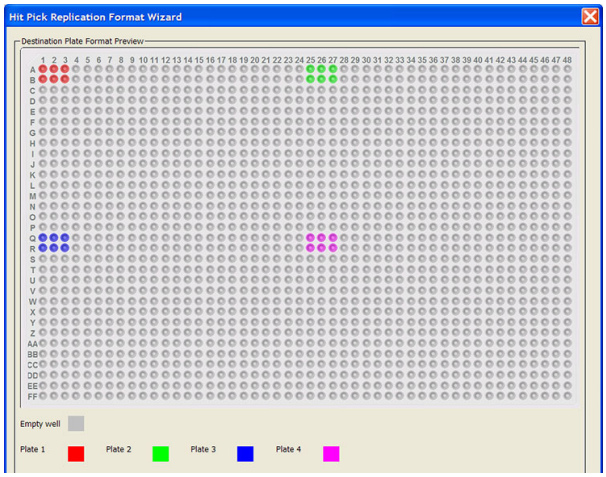
a In the **Operation Type** area, select **Plate replication.**

b In the **Labware** area, select the **Source plate labware** and the **Destination plate labware.**

c Click **Next.**

3 On the transfer page, select the **Transfer mode.**



Transfer mode	Description
Interleaved	Contents of corresponding wells from different source microplates are placed consecutively in the destination microplate. For example, if you are transferring contents from four 384-well source microplates into a 1536-well destination microplate, the Interleaved transfer would result in the following destination microplate layout: 
Quadrants	Contents from different source microplates are placed in quadrants of the destination microplate. For example, if you are transferring contents from four 384-well source microplates into a 1536-well microplate, the Quadrant transfer will result in the following destination microplate layout: 

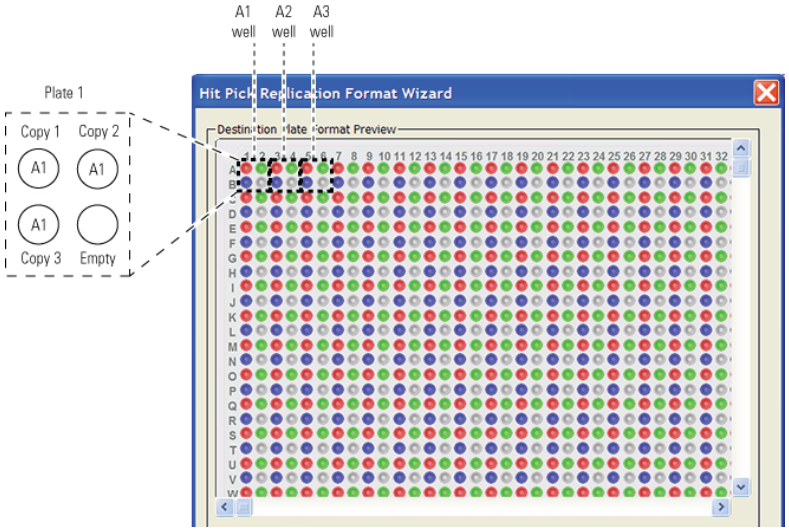
4 Type the volume to transfer from each source microplate well.

5 Type the number of copies you want to transfer from the source microplate.

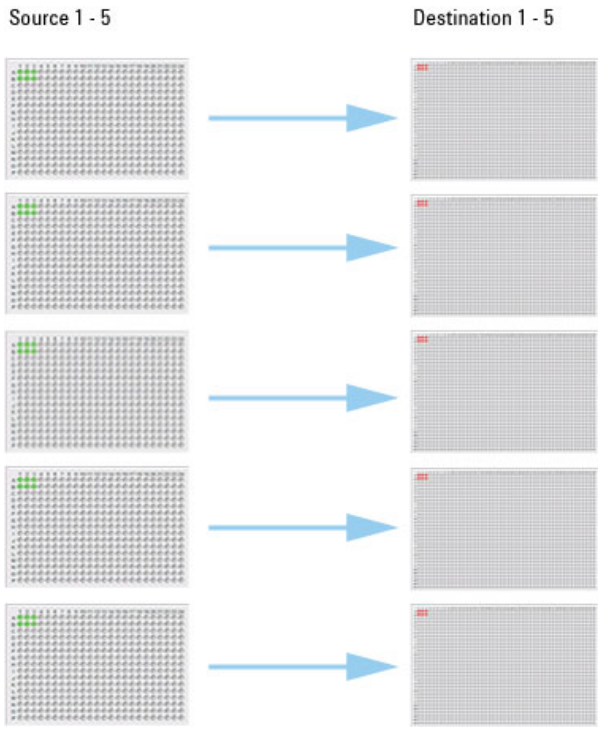
IMPORTANT

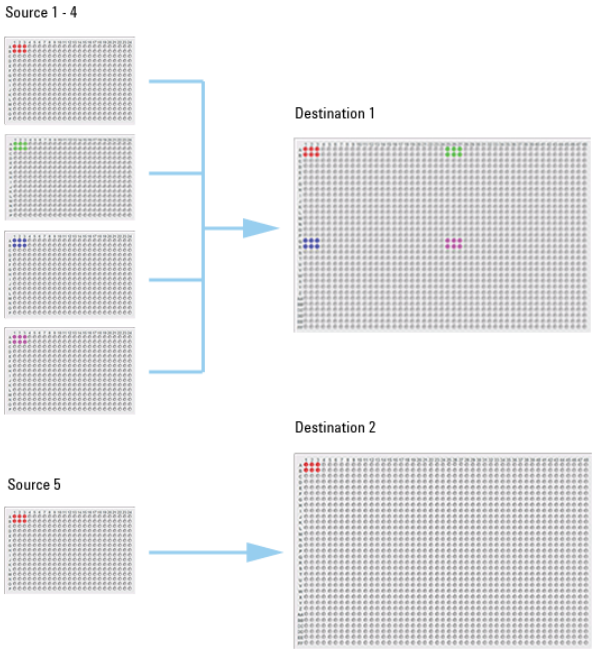
This value will affect the layout of the contents in the destination microplate.

For example, if you are transferring three copies of the contents from a 384-well source microplate into a 1536-well destination microplate, the Interleaved transfer will result in the following destination microplate layout. Notice that the wells in the fourth quadrant remain empty.



6 Select the destination microplate format:

Transfer mode	Description
All destination plates will have identical format/ patterning	All destination microplates will look identical. The layout of the destination microplate depends on the format of the source microplate, the number of copies you want to transfer, the total number of source microplates, and other selections in the wizard. For example, suppose you want to transfer contents from five 384-well source microplates into one or more 1536-well destination microplates. One 1536-well microplate can only accommodate contents from four 384-well microplates. To maintain the same layout across all of the destination microplates, five destination microplates will be used. The contents from each source microplate are transferred to one 1536-well destination microplate. *Note:* In the example, five destination microplates are used. 

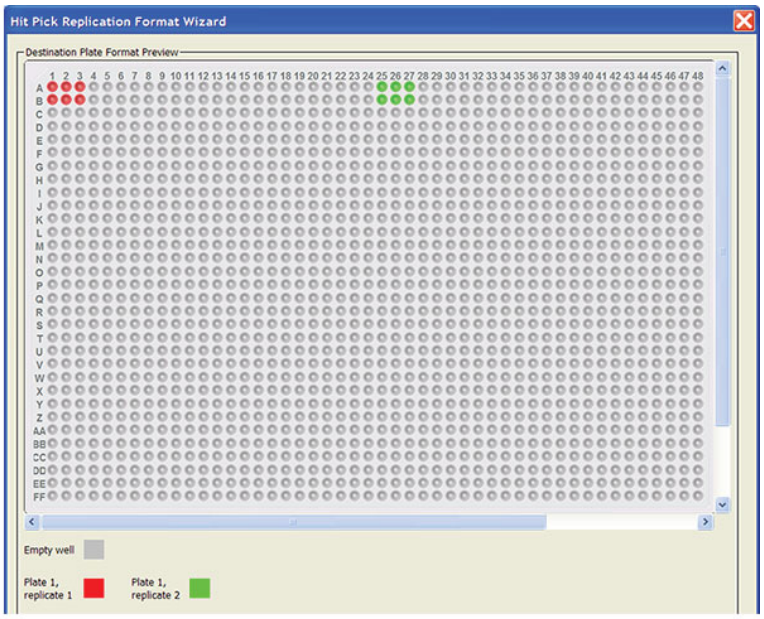
Transfer mode	Description
Minimize the number of destination plates used	The destination microplate will contain as many source contents as allowable to minimize the number of destination microplates used. For example, suppose you want to transfer contents from five 384-well source microplates into one or more 1536-well destination microplates. One 1536-well microplate can accommodate contents from four 384-well microplates. The contents from the fifth source microplate will be transferred into a second 1536-well destination microplate. *Note:* In this example, only two destination microplates are used. 

7 Select **Don't place replicates of the source plates on the same destination** if you want to place replicates in different destination microplates. Do not select the option if you want to place replicates in the same destination microplate.

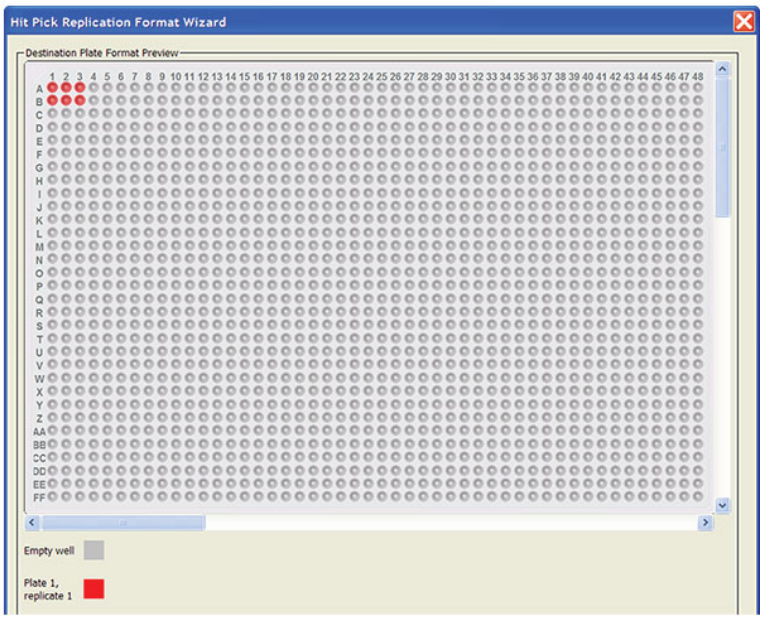
In the following example, the Quadrant transfer mode is used and the **All destination plates will have identical format/patterning option** is selected. Two replicates are requested from each source microplate.

In example a, the two replicates are placed in the same destination microplate. In example b, only one copy is placed in each destination microplate.

a)



b)



8 In the **Select Source Wells to Transfer** area, select the source wells from which to transfer the contents.

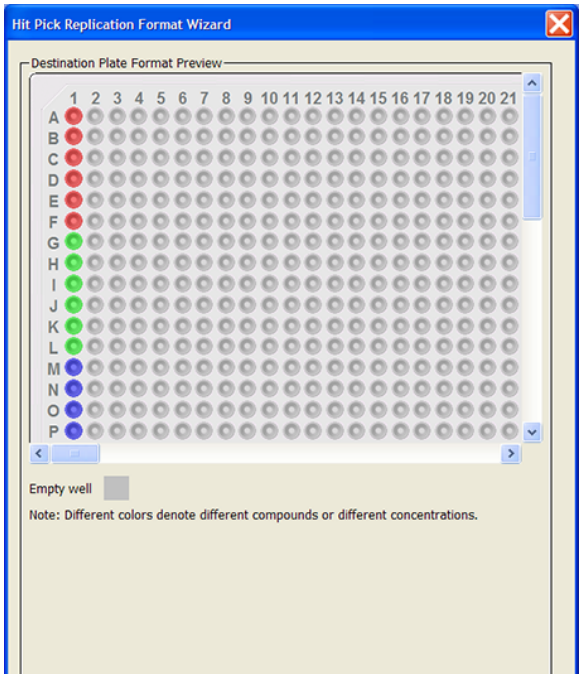
9 Click **Next** to preview the layout of the first destination microplate. The wells are color coded so you can see the replication pattern in the microplate.

To see the layout of all of the destination microplates, compile the protocol and check the output file. Alternatively, you can run the protocol in simulation mode and check the Pipette Log. For more information, see "Verifying source-destination transfers" on page 556.

Note that the layout shown in the preview might differ from the actual layout at run time if:

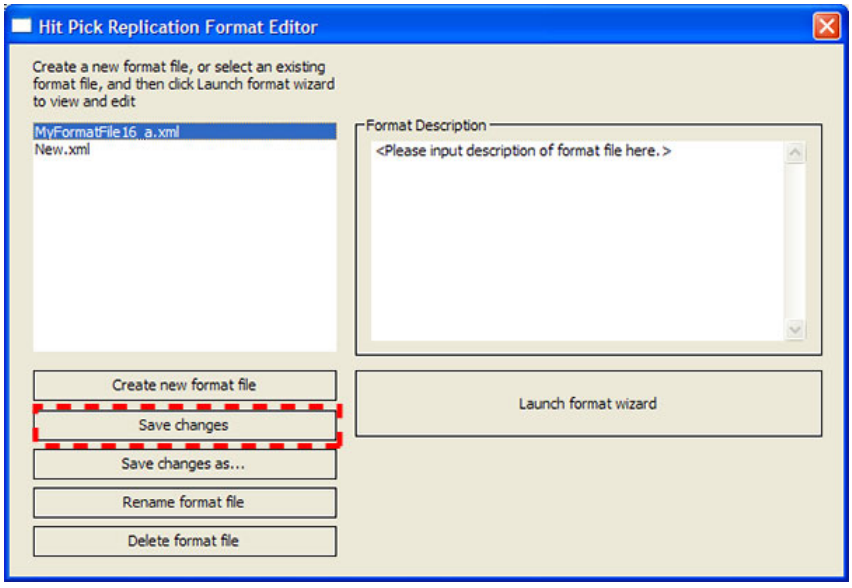
- The destination microplate has more wells than the source microplate.
- More than four source microplates will be processed during the run.

- You have selected **All destination plates will have identical format/patterning** in step 6.
- You have cleared the **Don't place replicates of source plates on the same destination** check box in step step 7.



When you are done, click **Finish.**

10 Back in the **Hit Pick Replication Format Editor,** make sure the correct format file name is selected, and then click **Save changes.** If you altered an existing format file, you can click **Save changes as** to save it as a different file.



11 Click the close button ( ) at the top right corner of the dialog box to return to the Hit Pick Routine Wizard. To continue in the Hit Pick Routine Wizard, return to step 3 in "Adding the Hit Pick Replication task" on page 530.

Verifying source-destination transfers

Before you start an actual run, you can verify that the source-destination transfers are correct using either output files or the Pipette Log.

Using output files

Compile your protocol to generate output files. The files list source-destination transfers by well and show the volume transferred. One output file is generated per source-destination microplate combination. For example, if contents from Source 1 are transferred to Destinations 1 and 2, then two output files are created, one for Source 1/Destination 1, and another for Source 1/Destination 2.

The following example shows an output file displayed in Excel. The first row shows the source and microplate IDs. Subsequent rows show the source wells, the corresponding destination wells, and the volume transferred.

	A	B	C
1	Plate2	1	Volume
2	Source Well	Destination Well	Volume in µL
3	C6	E1	2.5
4	C6	F1	2.5
5	B12	G1	2.5
6	B12	H1	2.5
7	M1	I1	2.5
8	M1	J1	2.5

Output files are in the CSV format. You set the directory for the hitpick output files in the Options dialog box, for example:

...\VWorks Workspace\VWorks\Hit Picking\Output Files\

For details, see "Viewing and setting file directory paths" on page 299.

Using the Pipette Log

Run your protocol in simulation mode and check the Pipette Log in the log area. The Pipette Log records every pipetting event that occurs, including aspirate location, dispense location, and volume information. For details, see "Pipette Log contents" on page 719.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Creating a protocol	• "Creating a protocol: basic procedure" on page 17
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Dilute to Final Volume task	"Dilute to Final Volume (Agilent Bravo)" on page 504
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514

For information about...	See...
Mix task	"Mix (Agilent Bravo)" on page 559
Pipette Log	"Pipette Log contents" on page 719
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Initialize Axis (Agilent Bravo)

Description

The Initialize Axis (Agilent Bravo) (

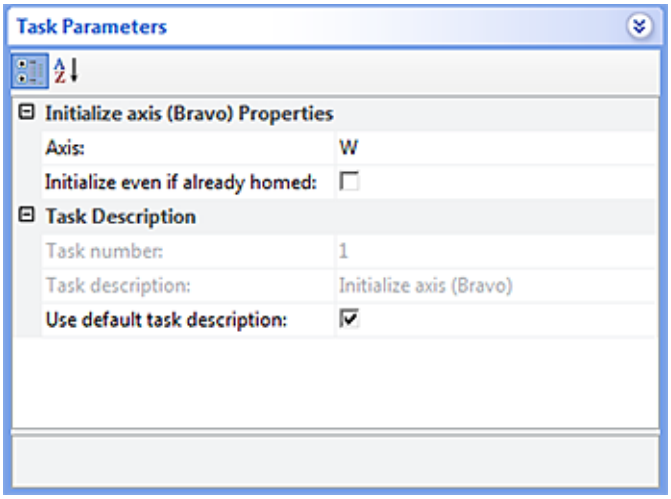
) task allows any one of the six Bravo axes (motors) to be initialized (homed) during a protocol run.

Task is available for...	Task is available in...
Bravo Platform	Startup Protocol Main Protocol, Bravo Subprocess Cleanup Protocol

For example, you could use this task with the AssayMAP Bravo Platform, which may contain liquid in the head syringes. You would initialize the device without initializing the *w*-axis to prevent dispensing any liquid onto a random deck location. Next, you would run a protocol to move the head to the waste location and then use the Initialize axis task to home the *w*-axis. For details, contact Agilent Agilent Technical Support.

Task parameters

After adding the Initialize axis task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Axis	The axis to be initialized. The options are W, X, Y, Z, Zg, and G.

Parameter	Description
Initialize even if already homed	The option to check if the Bravo axis is already homed before initializing it. If the check box is selected, no check is made to see if the Bravo axis is already homed before initializing (homing) the axis. If the check box is cleared, the Bravo is queried as to whether the axis is homed. • If the axis is already homed, the task completes without doing anything else. • If the axis is not already homed, the Bravo initializes (homes) the specified axis.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67

Mix (Agilent Bravo)

Description

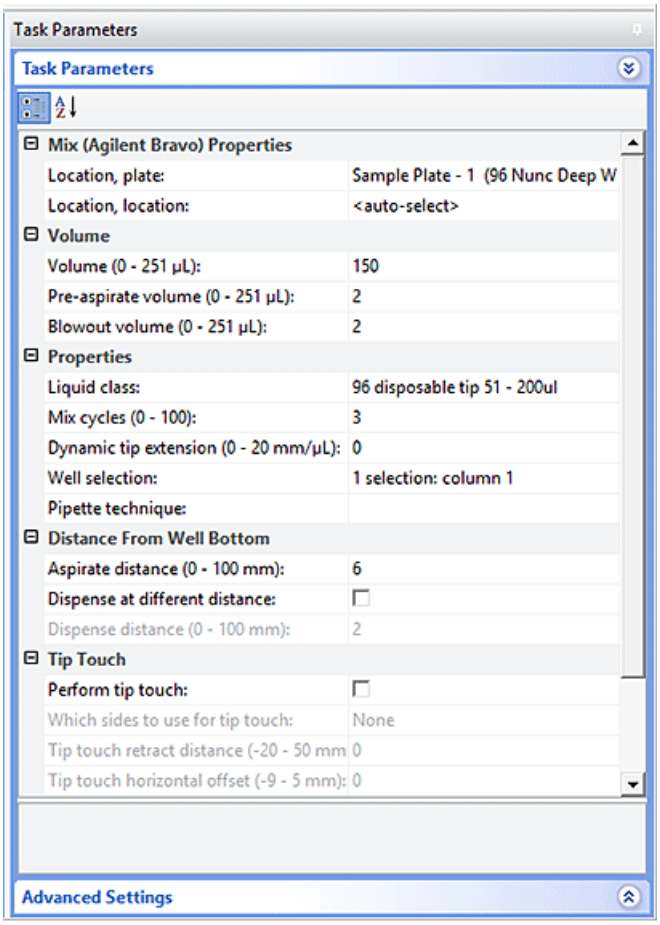
The Mix (Agilent Bravo) (

) task aspirates and dispenses liquid multiple times to mix it. The task allows you to specify different well-bottom distances for the aspirate and dispense actions.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Task parameters

After adding the Mix task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Volume (0–251 µL)	The volume of liquid to be mixed in each well.
Pre-aspirate volume (0–251 µL)	The volume of air to be drawn before the pipette tips enter the liquid.
Blowout volume (0–251 µL)	Specifies the volume of air to dispense after the main volume has been dispensed while the tips are still in the wells. Typically, the blowout volume is the same as the pre-aspirate volume.
Liquid class	The pipetting speed and accuracy. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Mix cycles ((0–100)	Specifies how many times to repeat the aspirate-and-dispense cycle.

Parameter	Description
Dynamic tip extension (0–20 mm/µL)	The rate at which the pipette head moves during the Aspirate task. The software calculates the distance over which the tips will move without crashing. Use dynamic tip extension to prevent spills as the pipette tips displace the liquid. To move the tips: • *At the same rate as the volume change*. Calculate dynamic tip extension (DTE) as follows: DTE = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTE > 1/A • *Slower than the volume change*. DTE < 1/A The starting and ending positions can be calculated as follows: $(V_{aspirated} * DTE) + Distance_{well\ bottom}$ *Note:* Instead of a negative aspirated volume, the software automatically moves downward toward the well bottom with each aspirate action.
Pipette technique	The pipette location offset you want to use for the Dispense task. The list of pipette techniques are defined in the Pipette Technique Editor.
Perform tip touch	The option to touch the pipette tip on one or more sides of the well.
Which sides to use for tip touch	The side or sides of the well to use during tip touch: North, South, East, West, North/South, West/East, West/East/South/North.
Tip touch retract distance (–20 to 50 mm)	The vertical distance for the pipette tips to rise before touching the sides of the wells.

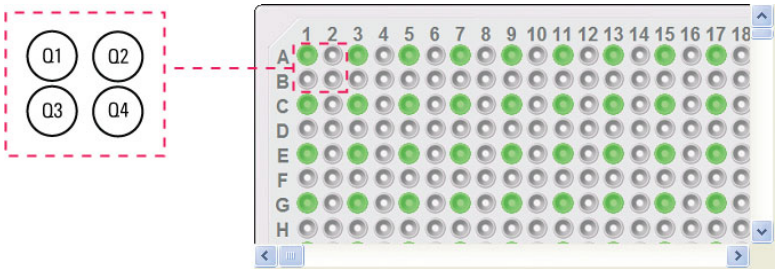
Parameter	Description
Tip touch horizontal offset (–9 to 5 mm)	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. For example, if you set a value of: • 0, the tips move a horizontal distance equal to the well radius • > 0, the tips attempt to move past the well radius, which results in a more forceful tip touch • < 0, the tips move a distance less than the radius of the well, resulting in a lighter tip touch

Quadrant pattern well selection

A quadrant is an evenly spaced array of locations that are accessible by the tips on a pipette head. The following table lists the types of pipette heads and the number of accessible quadrants in various microplates.

Pipette head channels/ pin tool pins	Microplate	Number of quadrants
96	96-well	1
	384-well	4
	1536-well	16
384	384-well	1
	1536-well	4
1536 (pin tool only)	1536-well	1

The following diagram demonstrates the concept of quadrants. The diagram shows a portion of a 384-well microplate and highlights the four quadrants (Q1, Q2, Q3, and Q4) that are accessible by the A1 tip of a 96-channel pipette head. Notice that the green color highlights all of the quadrant 1 (Q1) wells across the microplate.



Instead of a column- or row-wise pattern, you can select a quadrant pattern during well selection.

The quadrant pattern option is available only if:

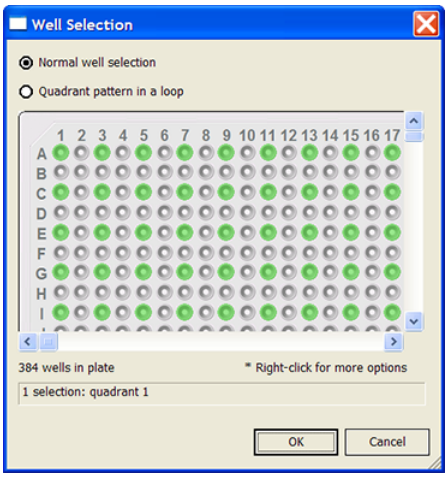
- The number of channels in the pipette head (or pins in a pin tool) is fewer than the number of wells in the microplate. For example, you can use a 96-channel pipette head to dispense liquid into a 384-well microplate or 1536-well microplate.
- All the channels are selected in the Set Head Mode task when using a pipette head. (The Set Head Mode task is not an option when using a pin tool).
- The liquid-handling task is inside a loop.

IMPORTANT

If you select a quadrant pattern, specifications in the Well Selection dialog box will override `task.Wellselection` values assigned in the Advanced Settings area.

To select a quadrant pattern:

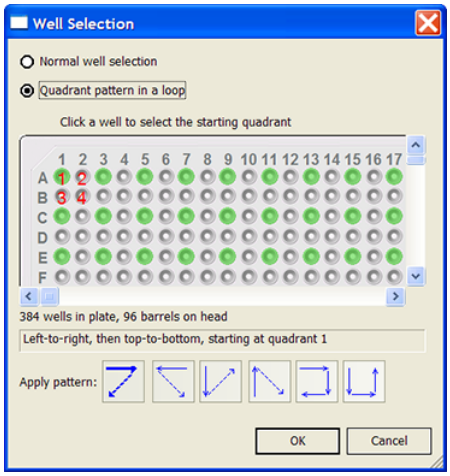
1 In the **Task Parameters** area, click the **Well selection** parameter box, and then click the Browse button. The Well Selection dialog box opens. By default, the **Normal well selection** option is selected. This option is used for column- and row-wise liquid-handling patterns.



2 Select **Quadrant pattern in a loop.** The contents of the dialog box change. Notice the following:

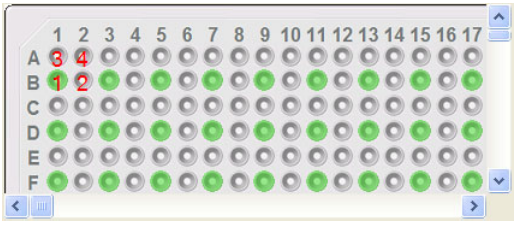
- Red numbers (1 through 4) appear on wells A1, A2, B1, and B2. The numbers indicate the pipetting sequence: 1 is the starting well, and 4 is the last well. In the following example, the sequence is A1, A2, B1, B2.
- Green wells indicate the starting well in the pipetting sequence.
- Pattern buttons at the bottom of the dialog box indicate the movement of the pipette channels. The movement description is provided in the text box above the buttons.

  Note: The last two patterns are unavailable if a group contains 16 wells. For example, the last two patterns are not available if you have a 96-well pipette head and a 1536-well microplate.



3 Select the starting well. The well becomes green and is labeled 1.

In the following example, the third quadrant (B1 well) is selected.



4 Click a pattern button to specify the pipette channel movement. After you click a pattern, the red numbers in the graphic are updated to show the sequence.

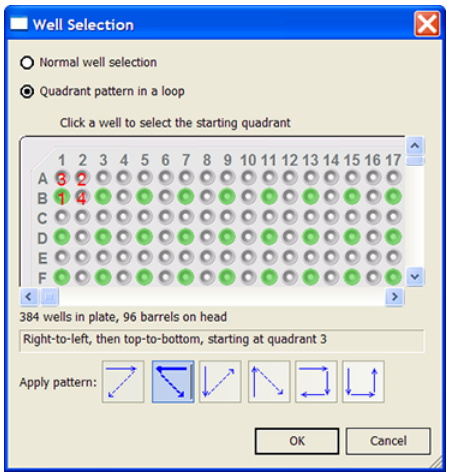
In the following example, the second pattern is selected (right-to-left, then top-to-bottom). The third quadrant (B1) is the starting well. The resulting movement is:

Quadrant 3 (B1)

Quadrant 2 (A2)

Quadrant 1 (A1)

Quadrant 4 (B2)



5 When you are finished, click **OK** to save the changes and return to the VWorks window.

Example: Mix the contents in the destination microplate on the Bravo Platform

Goal

Aspirate contents from a source microplate (Source 1), dispense into a destination microplate, and then mix the contents in the destination microplate. Use the default Mix parameters.

Implementation

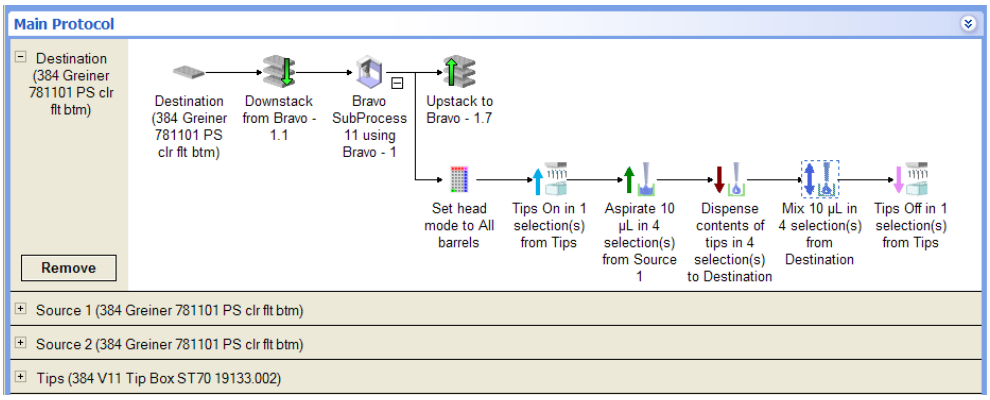
The Bravo deck is physically set up as follows:

- The destination microplates are at Bravo deck location 1.
- The source microplate is at deck location 2.
- The tip box is at deck location 9.

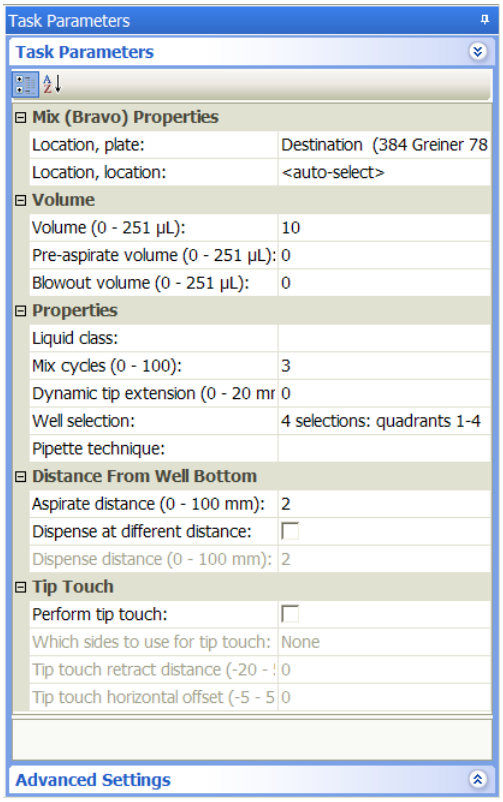
In the protocol, the following are added:

- Process for the destination microplate
- Configured labware for the source microplate
- Configured labware for the tip box

In the Destination plate process, a Bravo subprocess is added. Within the subprocess, a Mix task is added after the Aspirate and Dispense tasks as shown in the following example.



In the Mix Task Parameters area, Destination is selected, because the goal is to mix the contents in the destination microplate.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514
Set Head mode task	"Set Head Mode (Agilent Bravo)" on page 601
Tips On task	"Tips On (Agilent Bravo)" on page 623
Tips Off task	"Tips Off (Agilent Bravo)" on page 619
Pipette technique	"Specifying pipetting techniques" on page 691
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Move and Filter Plate (Agilent Bravo)

Description

The Move and Filter Plate task moves a specified filter plate from its current location to the Vacuum Filtration Station and then turns on and turns off the vacuum.

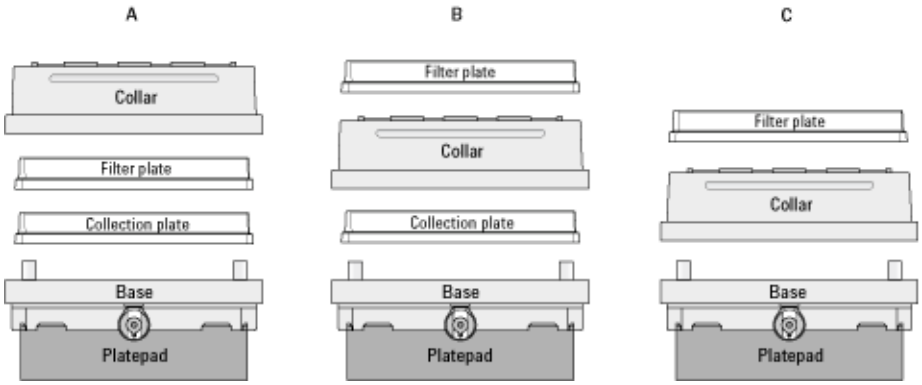
Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

In a Bravo Subprocess, use the Move and Filter Plate (Bravo) task (

) if the Vacuum Filtration Station is assembled in configuration B or C, as the following figure shows. In these configurations, the filter plate is not part of the station assembly.

Note: You may use the Move and Filter Plate task with configuration A if you want to place a second filter plate on top of configuration A.

Figure Vacuum Filtration Station configurations A, B, and C

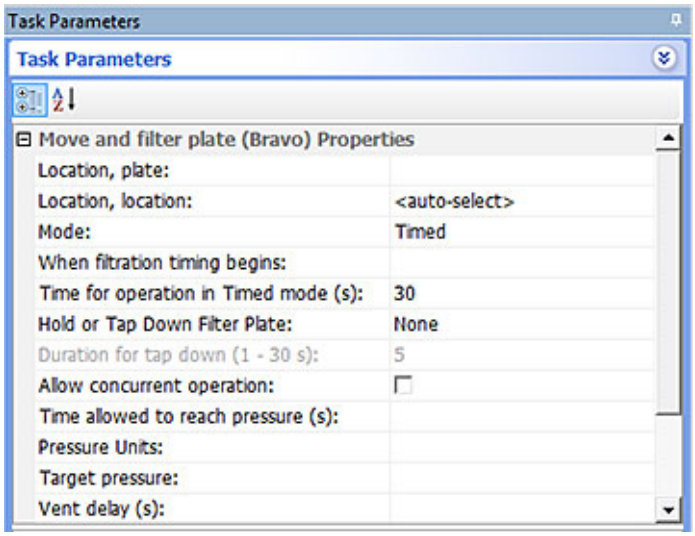


If the Vacuum Filtration Station has configuration A, where the filter plate is part of the assembly (it sits under the collar), use the Toggle Vacuum task to turn on and turn off the vacuum. See "Toggle Vacuum (Agilent Bravo)" on page 626.

Task parameters

Note: The Move and Filter task parameters displayed can vary, depending on the type of vacuum pump used with the Vacuum Filtration Station.

After adding the Move and Filter Plate task at the desired point in the Bravo Subprocess, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The name of the filter plate that the robot will move to the Vacuum Filtration Station (specified by the Location, location parameter). Select the filter plate from the list of names. *Note:* In a double-filtration scenario, where a Move and Filter Plate task will place a filter plate on top of configuration A, ensure that you specify the filter plate to be placed on top of the collar, not the filter plate that is part of the configuration A assembly.
Location, location	The deck location of the Vacuum Filtration Station. If more than one Vacuum Filtration Station is on the deck, select the specific deck location instead of <auto-select>.
Mode	The action of the task. The options are On, Off, and Timed. • **On and Off**. If you are not timing the filtering process, add two Move and Filter Plate tasks in the protocol for each filtering operation. One task turns on the vacuum (Mode = On), and the other task turns off the vacuum (Mode = Off). • **Timed**. If you plan to time the filtering process, add only one Move and Filter Plate task. The task turns on the vacuum, and then turns off the vacuum automatically at the end of the time period.

Parameter	Description
When filtration timing begins *ME4C VARIO Vacuum Pump only*	The different options for when to start timing the filtration process: • When pressure is achieved • When the vacuum pump starts
Time for operation in Timed mode (s)	The duration, in seconds, that you want to leave the vacuum on. At the end of the period, the vacuum will turn off.
Hold or tap down filter plate	The different options for whether to have the Bravo gripper hold down the filter plate when the vacuum is turned on to ensure a secure vacuum seal: • **None**. The Bravo gripper will not hold down the filter plate. • **Tap down**. The Bravo gripper will hold down the filter plate from the top only for the time period specified in the Duration for tap down parameter. • **Hold down**. The Bravo gripper will hold down the filter plate from the top for the duration of the task. *Note:* If the Vacuum Filtration Station is assembled in configuration A without a filter plate on top of the collar, and the Tap down or Hold down option is selected, the gripper will press directly on the outer upper plastic rim of the collar. *Note:* If you select None or Tap down, the protocol can perform other tasks in parallel. Concurrent operation is not an option if you select Hold down.
Duration for tap down (1-30 s)	The length of time, in seconds, that the Bravo gripper presses down on the filter plate. The timing commences the instant that the gripper begins to press down on the filter plate, not from the commencement of the vacuum. *Note:* The Vario vacuum pump has a delay of approximately 3 seconds to start the vacuum after the Tap down begins.
Allow concurrent operation	The option to permit the accessory to operate simultaneously with other tasks.
Time allowed to reach pressure (s) *ME4C VARIO Vacuum Pump only*	The length of time, in seconds, to allow the vacuum to reach the specified target pressure. An error message displays if the target pressure is not reached within the time specified.

Parameter	Description
Pressure units *ME4C VARIO Vacuum Pump only*	The desired unit of measure: mbar, Torr, hPa, mmHg, cmHg, or inHg. **IMPORTANT** The unit of measure specified in the Bravo device profile must match the pressure unit set at the VARIO pump in order for the VWorks software to convert to the selected units.
Target pressure *ME4C VARIO Vacuum Pump only*	The difference between the pressure of the outside atmosphere above the filter and the pressure in the Vacuum Filtration Station manifold, including the enclosure beneath the filter. For example, if you set the Target pressure to 600 mbar and the ambient pressure displayed on the VARIO pump is 1000 mbar, the vacuum will remain on until the reading on the VARIO pump reaches 400 mbar.
Vent delay *ME4C VARIO Vacuum Pump only*	The length of time, in seconds, to wait for the air pressure under the filter to equalize with the ambient air pressure.

Example

Goal

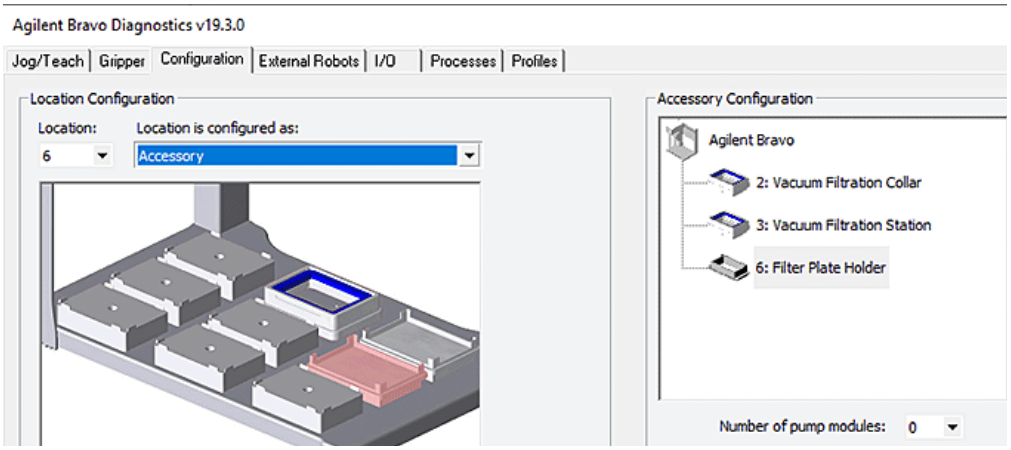
Assemble a Vacuum Filtration Station whose configuration is Base–Collection plate–Collar. Move a filter plate to the station. Filter for 30 seconds. Disassemble the station.

Implementation

In Bravo Diagnostics, the Bravo deck is configured such that:

- The Vacuum Filtration Station will be assembled at deck location 3. The Vacuum Filtration Station base is at deck location 3.
- The collar is at deck location 2.
- The Filter Plate Holder is at deck location 6.

Figure Configuration tab in the Bravo Diagnostics dialog box



For instructions on how to configure the device in Bravo Diagnostics, see the *Bravo Platform User Guide*.

In the protocol, the following are added as shown:

- Filter Plate process
- Collection Plate configured labware at deck location 8
- Place Plate (to place the Filter Plate at location 6), Assemble Vacuum, Move and Filter Plate, and Disassemble Vacuum tasks

Figure Example of protocol with the Move and Filter Plate (Bravo) task

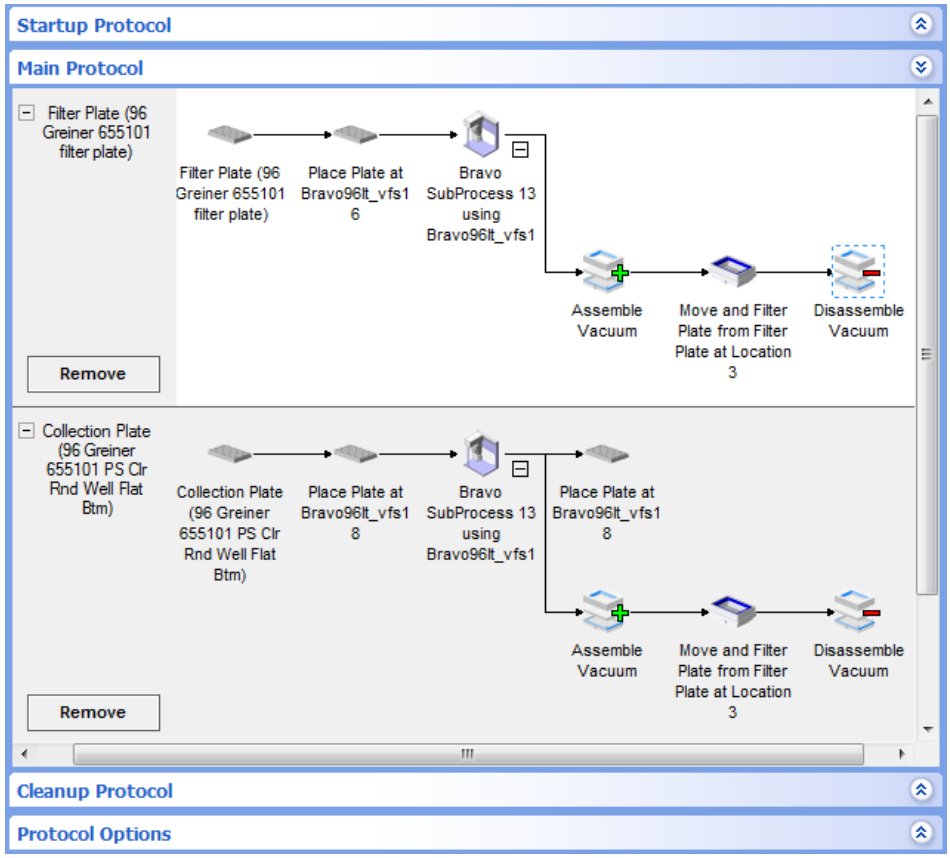


Figure Assemble Vacuum task parameters example

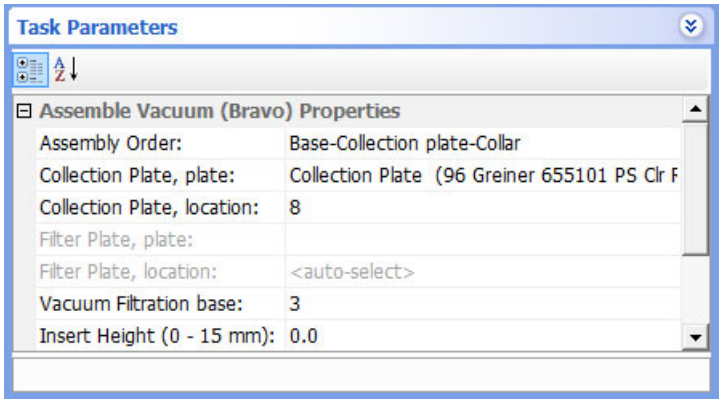


Figure Move and Filter Plate task parameters example

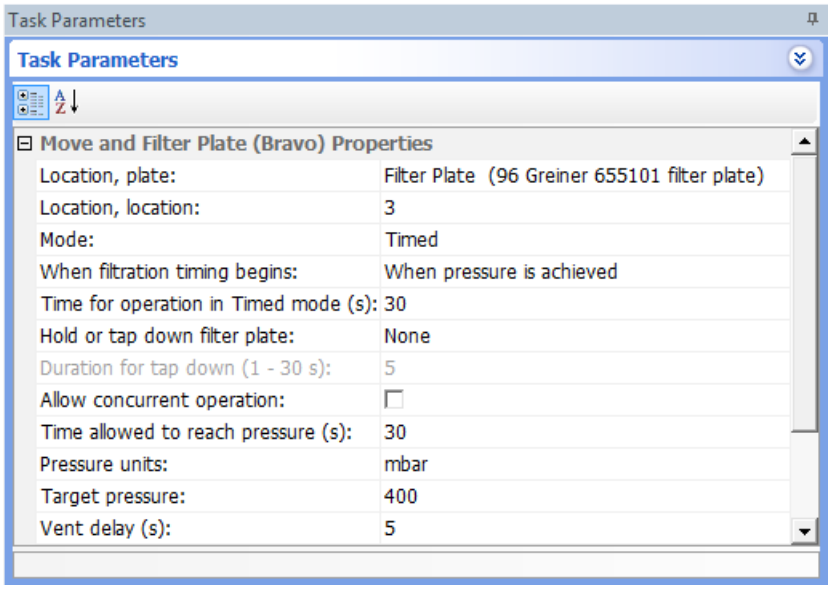
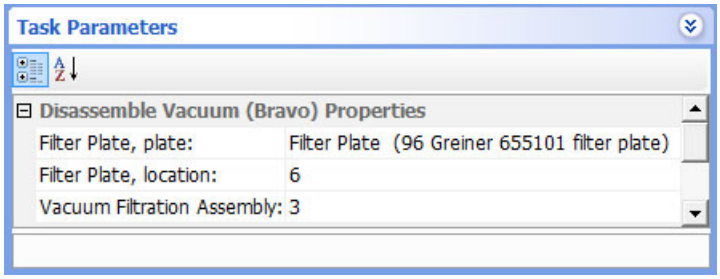


Figure Disassemble Vacuum task parameters example



The resulting protocol will run as follows:

1. Acknowledge that the Filter Plate is starting at deck location 6.
2. Acknowledge that the Collection Plate is starting at deck location 8.
3. Assemble the Vacuum Filtration Station (Base–Collection plate–Collar):
   a. Move the Collection Plate from deck location 8 and place it on top of the base at deck location 3.

b Move the collar from deck location 2 and place it on top of the Collection Plate at deck location 3.

4 Move the Filter Plate from deck location 6 and place it on top of the assembled Vacuum Filtration Station at deck location 3.

5 Turn on the vacuum for 30 seconds, and then turn off the vacuum.

6 Disassemble the Vacuum Filtration Station:

a Move the Filter Plate to deck location 6.

b Move the collar back to deck location 2.

c Move the Collection Plate to deck location 8.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Assemble Vacuum task	"Assemble Vacuum (Agilent Bravo)" on page 499
Disassemble Vacuum task	"Disassemble Vacuum (Agilent Bravo)" on page 512
Toggle Vacuum task	"Toggle Vacuum (Agilent Bravo)" on page 626
Pipetting in a filter plate that sits under the collar	"Pipette Under Collar Begin and End (Agilent Bravo)" on page 584
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Pin Tool (Agilent Bravo)

Description

The Pin Tool (Agilent Bravo) (

) task can be used to perform low-volume transfers of a fixed volume using a pin tool. You can use the Pin Tool task repeatedly in a protocol subprocess to perform all the pin-tool-related steps, such as:

- Pin Tool—Adsorb

- Pin Tool—Dispense
- Pin Tool—Wash
- Pin Tool—Blot
- Pin Tool—Mix

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Before you begin

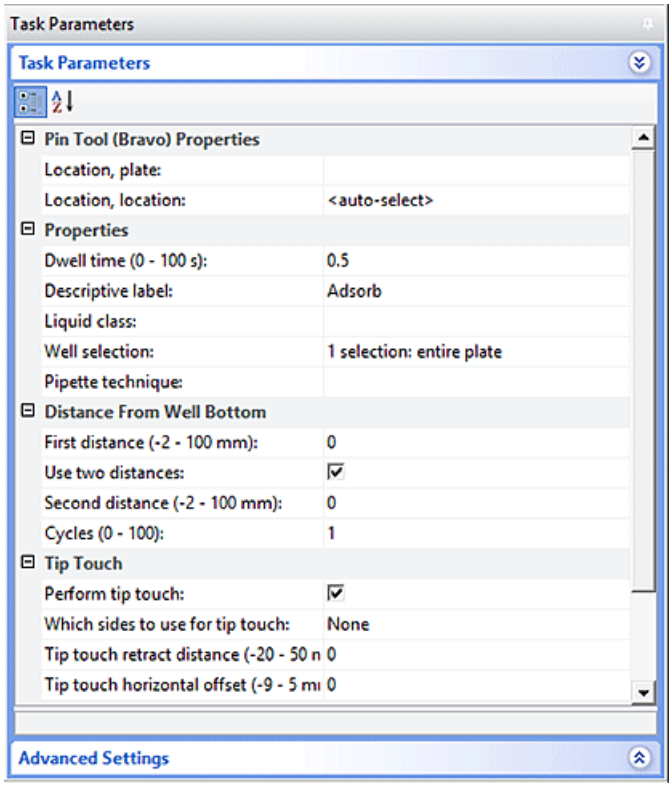
Ensure the following:

- The Bravo device profile specifies an appropriate pin tool. To create or edit a profile, see the user guide for the applicable device.

  Note: Ensure the pin tool teachpoints are set up in the same manner as a fixed-tip pipette head.
- The protocol includes the SubProcess (Bravo).

Task parameters

After adding the Pin Tool task at the desired point in the subprocess, set the following parameters in the **Task Parameters** area:



Parameter	Description
Dwell time (s)	The time duration that the pins remain at the specified height (First distance or Second distance) within the well. For example, you might start with the following values: • Adsorb, Dispense into fluid, or Mix—0.5 s or longer for more viscous fluids • Blot—2 s, or longer for more viscous fluids
Descriptive label	A text label that you can add to the task icon in the protocol. Click the arrow in the Descriptive label box to choose an option. The options include: • Enter a JavaScript variable or script. • Use a predefined label: Adsorb, Dispense, Wash, Blot, or Mix • Type your own label in the box.
Liquid class	A parameter that you can use to control the accuracy and the speed of the pin tool as it moves into and out of the wells. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Pipette technique	The pipette location offset you want to use for the Pin Tool task. The list of pipette techniques are defined in the Pipette Technique Editor.
First distance (mm)	The first height for the pin tool during the Pin Tool task. The value is the distance between the pin tips and the well bottoms. For example, during an adsorb step, you might set this value to 0 mm so that the pin tips touch the bottom of the wells. This parameter can affect the quantity adsorbed. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the pins at the correct distance from the well bottom.

Parameter	Description
Use two distances	The option to specify a second height for the pins during the Pin Tool task. For example, you could cycle the pin positions between two heights within the wells repeatedly to perform mixing or to wash the pins. Default: Not selected
Second distance (mm)	The distance between the pin tips and the well bottoms at the second height for the pins. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the pins at the correct distance from the well bottom.
Cycles	Available if you select the Use two distances option. The Cycles parameter sets the number of times to move the pins repeatedly to the two heights, for example to perform mixing or to wash the pins.
Perform tip touch	The option to touch the pins on one or more sides of the well, or to enable the pins to make lateral stirring moves inside the fluid, for example during a wash task.
Which sides to use for tip touch	The side or sides of the well to use during tip touch: North, South, East, West, North/South, West/East, West/East/South/North.
Tip touch retract distance (mm)	The vertical distance for the pins to move before moving laterally within the well, where • 0 is the vertical distance equal to the well bottom • > 0 is the vertical distance the pins rise above the bottom • < 0 is the vertical distance the pins attempt to move past the well bottom

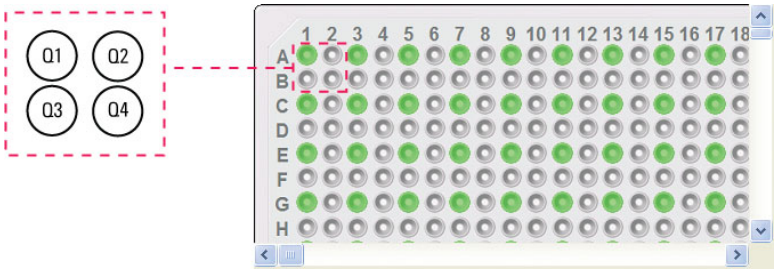
Parameter	Description
Tip touch horizontal offset (mm)	The horizontal distance that the pins move. The value is based on the well diameter specified by the labware definition, where • 0 is a distance equal to the well radius • > 0 is the distance the pins attempt to move past the well radius, which results in a more forceful tip touch • < 0 is a distance less than the radius of the well, resulting in a lighter tip touch or no tip touch

Quadrant pattern well selection

A quadrant is an evenly spaced array of locations that are accessible by the tips on a pipette head. The following table lists the types of pipette heads and the number of accessible quadrants in various microplates.

Pipette head channels/ pin tool pins	Microplate	Number of quadrants
96	96-well	1
	384-well	4
	1536-well	16
384	384-well	1
	1536-well	4
1536 (pin tool only)	1536-well	1

The following diagram demonstrates the concept of quadrants. The diagram shows a portion of a 384-well microplate and highlights the four quadrants (Q1, Q2, Q3, and Q4) that are accessible by the A1 tip of a 96-channel pipette head. Notice that the green color highlights all of the quadrant 1 (Q1) wells across the microplate.



Instead of a column- or row-wise pattern, you can select a quadrant pattern during well selection.

The quadrant pattern option is available only if:

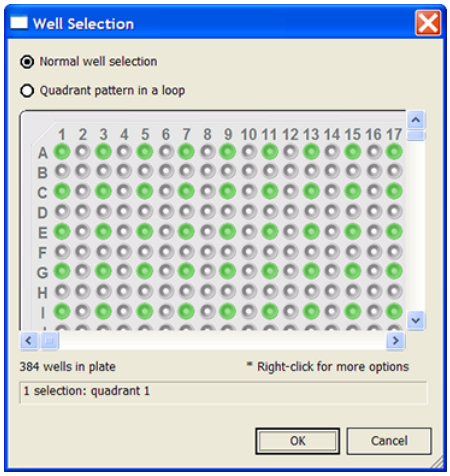
- The number of channels in the pipette head (or pins in a pin tool) is fewer than the number of wells in the microplate. For example, you can use a 96-channel pipette head to dispense liquid into a 384-well microplate or 1536-well microplate.
- All the channels are selected in the Set Head Mode task when using a pipette head. (The Set Head Mode task is not an option when using a pin tool).
- The liquid-handling task is inside a loop.

IMPORTANT

If you select a quadrant pattern, specifications in the Well Selection dialog box will override `task.Wellselection` values assigned in the Advanced Settings area.

To select a quadrant pattern:

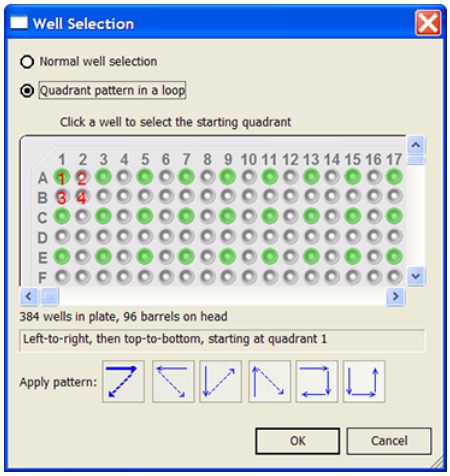
1 In the **Task Parameters** area, click the **Well selection** parameter box, and then click the Browse button. The Well Selection dialog box opens. By default, the **Normal well selection** option is selected. This option is used for column- and row-wise liquid-handling patterns.



2 Select **Quadrant pattern in a loop.** The contents of the dialog box change. Notice the following:

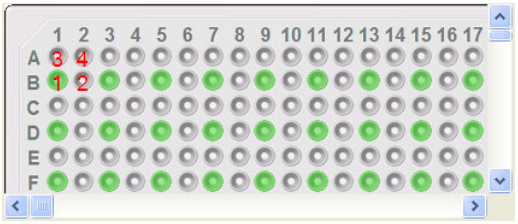
- Red numbers (1 through 4) appear on wells A1, A2, B1, and B2. The numbers indicate the pipetting sequence: 1 is the starting well, and 4 is the last well. In the following example, the sequence is A1, A2, B1, B2.
- Green wells indicate the starting well in the pipetting sequence.
- Pattern buttons at the bottom of the dialog box indicate the movement of the pipette channels. The movement description is provided in the text box above the buttons.

  Note: The last two patterns are unavailable if a group contains 16 wells. For example, the last two patterns are not available if you have a 96-well pipette head and a 1536-well microplate.



3 Select the starting well. The well becomes green and is labeled 1.

In the following example, the third quadrant (B1 well) is selected.



4 Click a pattern button to specify the pipette channel movement. After you click a pattern, the red numbers in the graphic are updated to show the sequence.

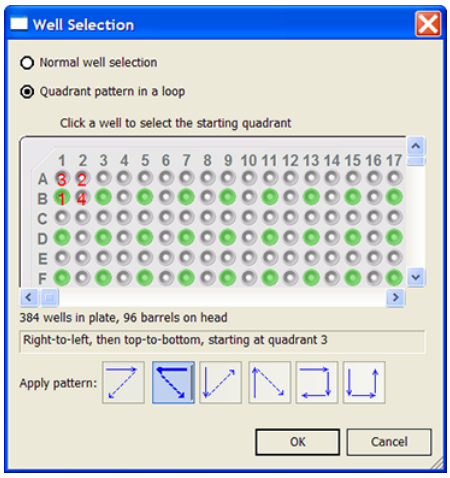
In the following example, the second pattern is selected (right-to-left, then top-to-bottom). The third quadrant (B1) is the starting well. The resulting movement is:

Quadrant 3 (B1)

Quadrant 2 (A2)

Quadrant 1 (A1)

Quadrant 4 (B2)



5 When you are finished, click **OK** to save the changes and return to the VWorks window.

Example: Pin Tool tasks on a Bravo Platform

Goal

Using a pin tool, transfer a small volume from a source microplate (Source 1) into a destination microplate, and wash and blot the pins.

Implementation

The Bravo Platform is physically set up as follows:

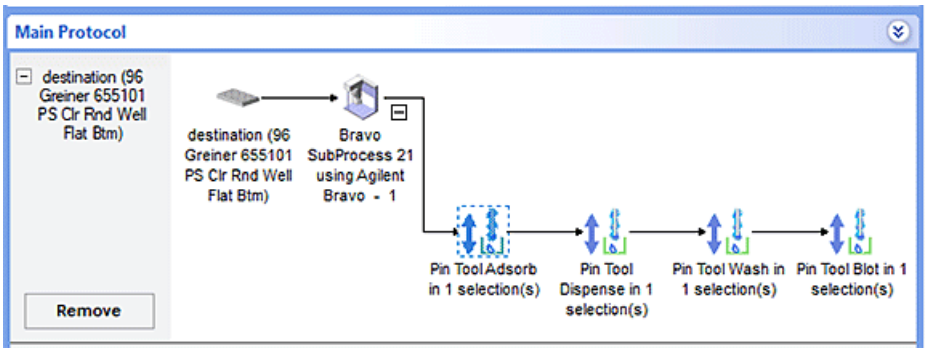
- The source microplate is on deck location 3.
- The destination microplate is on deck location 6.
- The wash station is on deck location 1.
- The blotting material is on deck location 5.

In the protocol, the following are added:

- Process for the destination microplate

- Configured labware for the source microplate
- Configured labware for the wash station
- Configured labware for the blotting station

In the process, a Bravo subprocess is added. Within the subprocess, four Pin Tool tasks are added for Adsorb, Dispense, Wash, and Blot.



Parameter settings for each Pin Tool task in the example

- Adsorb
  - Dwell time 0.5 s
  - First distance 0 mm
  - Second distance 10 mm
  - Cycles 3
  - No tip touch
- Dispense (into dry microplate)
  - Dwell time 0.5 s
  - First distance –2 mm
  - No second distance
  - No tip touch
- Wash
  - Dwell time 0.5 s
  - First distance 0 mm
  - Second distance 10 mm
  - Cycles 3
  - Perform tip-touch on all sides with horizontal offset of 0 mm and at a retract distance of 2 mm
- Blot
  - Dwell time 2 s
  - First distance –2 mm
  - No second distance
  - No tip touch

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Liquid classes	*VWorks Automation Control Setup Guide*
Set Head Mode task	"Set Head Mode (Agilent Bravo)" on page 601
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Pipette Under Collar Begin and End (Agilent Bravo)

Description

If the assembled Vacuum Filtration Station includes the filter plate, as shown in the following figure, and you want to pipette into the filter plate that is underneath the collar, a pair of Pipette Under Collar tasks (Begin and End) are required.

- **Pipette Under Collar Begin (Agilent Bravo)** (

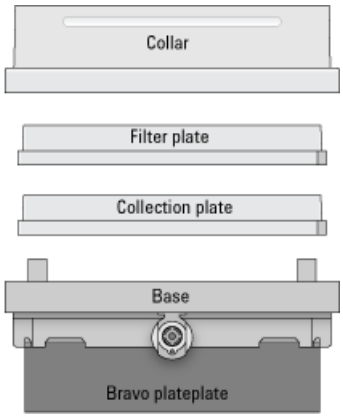
). Prepares the Bravo device for pipetting in a microplate that sits under the collar of the Vacuum Filtration Station.
- **Pipette Under Collar End (Agilent Bravo)** (

). Resets the Bravo device after pipetting under the Vacuum Filtration Station collar to prepare for subsequent tasks in the Bravo Subprocess.

CAUTION

If you are using the collar with the blue gasket, ensure that you remove the clear gasket frame support from the top of the collar. Otherwise, a collision can occur between the pipette tips and the gasket frame support.

Figure Vacuum Filtration Station configuration with the filter plate as part of the assembly



Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

IMPORTANT

When the protocol runs, an error message will appear before the Pipette Under Vacuum Collar Begin task starts. The operator must click OK to proceed. Alternatively, this error can be added to the Error Library.

Task parameters

Note: The Pipette Under Collar Begin task does not have any task parameters.

Pipetting task parameter requirements

The pipetting task (for example, Dispense) that is placed between the Pipette Under Collar Begin task and Pipette Under Collar End task has the following requirements:

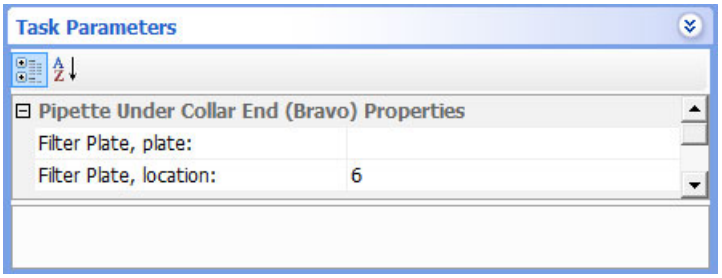
- The Location, plate parameter must specify the filter plate already underneath the collar of the Vacuum Filtration Station.

  Note: The Pipette Under Collar tasks are not used for configurations of the Vacuum Filtration Station that do not include the filter plate as part of the assembly.
- The Location, location parameter must specify the actual location of the Vacuum Filtration Station. Location, location cannot be set at <auto-select>.
- The Well selection must be for the entire plate.

CAUTION

A collision can occur if the Bravo head is partially offset from the collar of the Vacuum Filtration Station (partial head mode) while attempting to pipette into a filter plate that is under the collar. Ensure that the well selection is for the entire plate.

Pipette Under Vacuum Collar End task parameters



Parameter	Description
Filter Plate, plate	The name of the filter plate that is underneath the collar in the assembled Vacuum Filtration Station. Select the filter plate from the list of names.
Filter Plate, location	The deck location that the filter plate occupied immediately before the Assemble Vacuum task that assembled the Vacuum Filtration Station earlier in the Bravo Subprocess. For example, if the Assemble Vacuum task took the filter plate from a Filter Plate Holder at deck location 6, the Filter Plate, location value must be set to 6.

Example

Goal

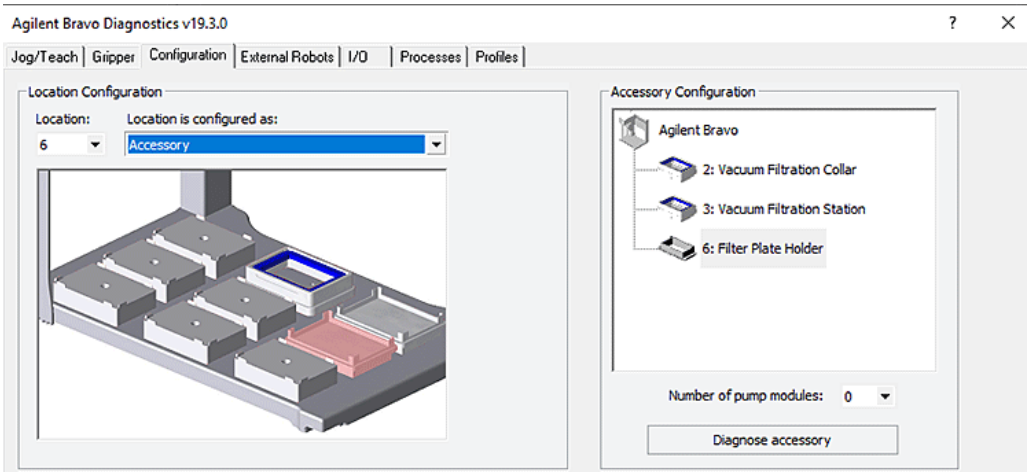
Assemble a Vacuum Filtration Station in the following configuration: Base–Collection plate–Filter plate–Collar). Aspirate liquid from a reservoir, dispense liquid into the Filter Plate under the collar. Filter for 60 seconds. Disassemble the station.

Implementation

In Bravo Diagnostics, the profile is configured such that:

- The Vacuum Filtration Station will be assembled at deck location 3.
- The Vacuum Filtration Station base is at deck location 3.
- The collar is at deck location 2.
- The Filter Plate Holder is at deck location 6.

Figure Configuration tab in Bravo Diagnostics dialog box

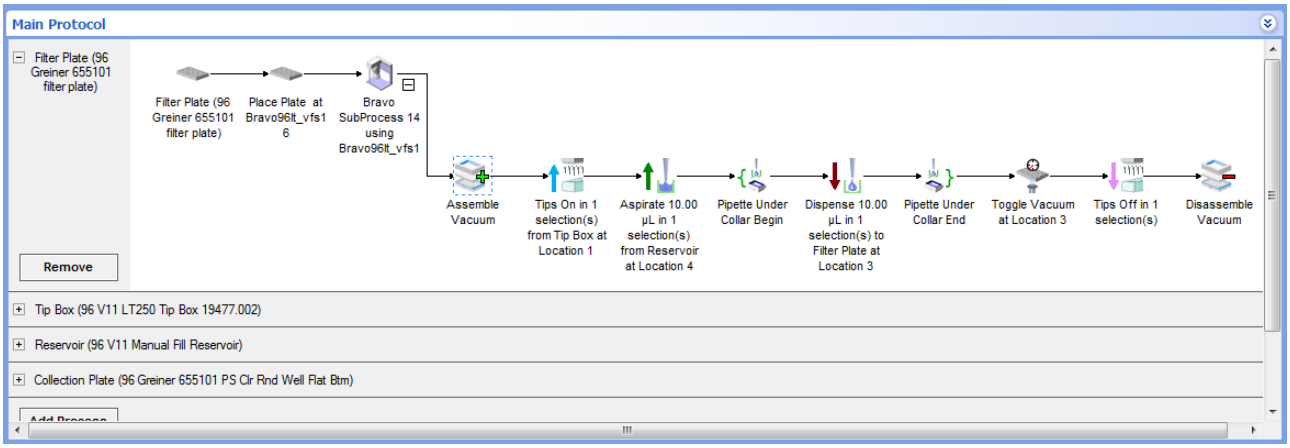


For instructions on how to configure the device in Bravo Diagnostics, see the *Bravo Platform User Guide*.

In the protocol, the following are added, as shown:

- Filter Plate process with a Place Plate task to place the Filter Plate at Bravo deck location 6.
- Configured labware for the tip box, reservoir, and Collection Plate.
- In the Bravo SubProcess, the following tasks are added: Assemble Vacuum, Tips On, Aspirate, Pipette Under Collar Begin, Dispense, Pipette Under Collar End, Toggle Vacuum, Tips Off, and Disassemble Vacuum.

Figure Example of protocol with Pipette Under Collar tasks



Note the following:

- Assemble Vacuum task Assembly order is for Base–Collection plate–Filter plate–Collar.
- Tips On task Well selection is for the entire plate.
- Dispense task specifies the Filter Plate that is under the collar in assembled Vacuum Filtration Station at deck location 3.
- Pipette Under Collar Begin and End tasks bracket the Dispense task.

The following figures show some of the task parameter settings.

Figure Assemble Vacuum task parameters

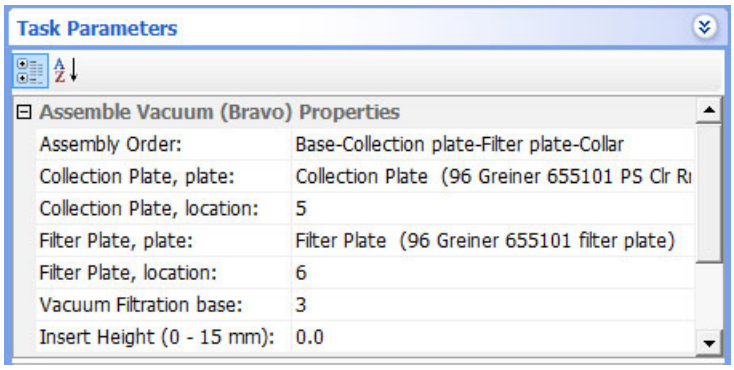


Figure Tips On task parameters

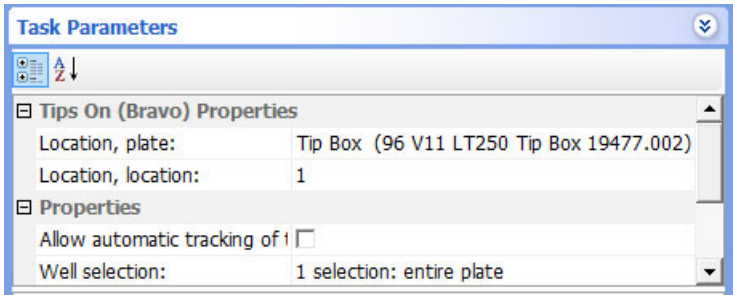


Figure Aspirate task parameters

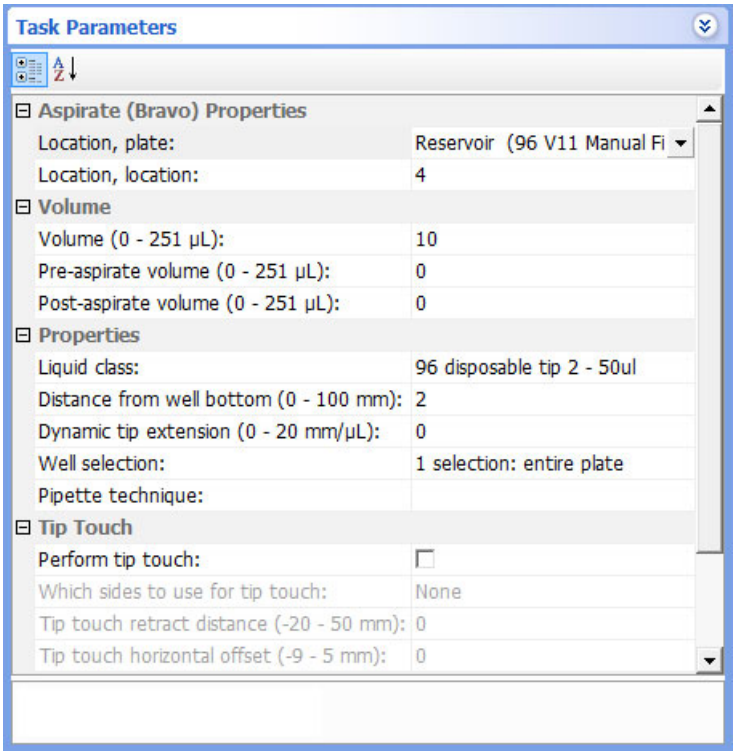


Figure Dispense task parameters

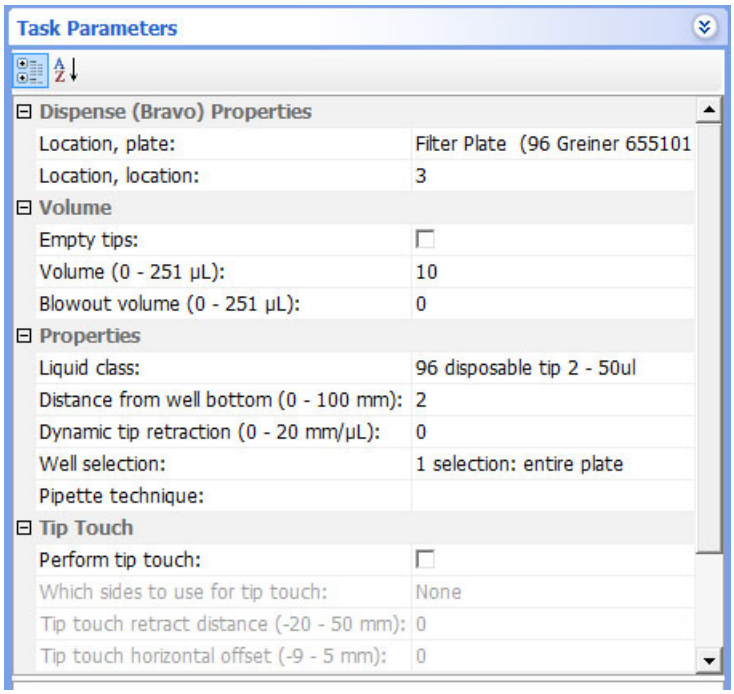


Figure Pipette Under Collar End parameters

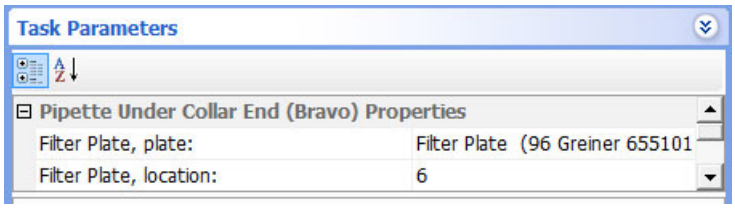


Figure Toggle Vacuum task parameters

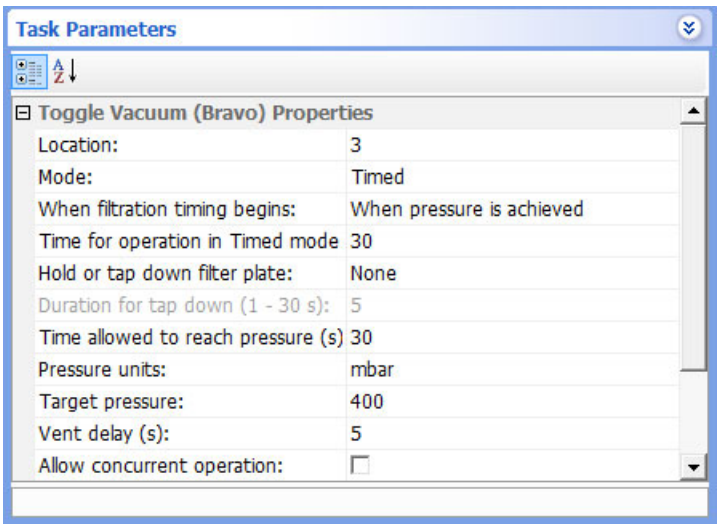
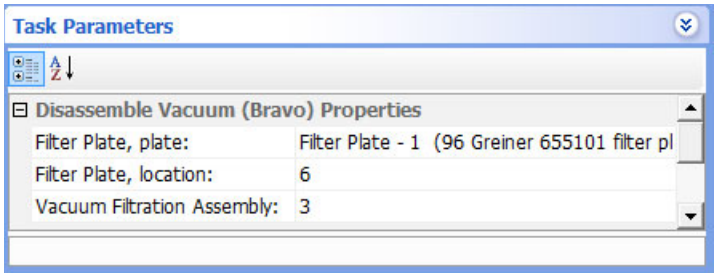


Figure Disassemble Vacuum task parameters



The resulting protocol will run as follows:

1. Acknowledge that the Filter Plate is starting at deck location 6.
2. Acknowledge that the Collection Plate is starting at deck location 5.
3. Assemble the Vacuum Filtration Station (Base–Collection plate–Filter plate–Collar) as follows:
   - **a** Move the Collection Plate from deck location 5 and place it atop the base at deck location 3.
   - **b** Move the Filter Plate from deck location 6 and place it atop the Collection Plate at deck location 3.
   - **c** Move the collar from deck location 2 and place it on top of the Filter Plate at deck location 3.
4. Prepare the Bravo device to pipette under the vacuum collar.

5 Dispense into the Filter Plate in the Vacuum Filtration Station at deck location 3.
6 Reset the Bravo device for tasks other than pipetting under the vacuum collar.
7 Turn on the vacuum for 30 seconds, and then turn off the vacuum.
8 Disassemble the Vacuum Filtration Station:
   a Move the collar back to deck location 2.
   b Move the Filter Plate back to deck location 6.
   c Move the Collection Plate back to deck location 5.

Related information

For information about...	See...
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Assemble Vacuum task	"Assemble Vacuum (Agilent Bravo)" on page 499
Disassemble Vacuum task	"Disassemble Vacuum (Agilent Bravo)" on page 512
Move and Filter Plate task	"Move and Filter Plate (Agilent Bravo)" on page 568
Toggle Vacuum task	"Toggle Vacuum (Agilent Bravo)" on page 626

Pump Reagent (Agilent Bravo)

Description

The Pump Reagent (Agilent Bravo) ( Pump Reagent (Agilent Bravo)) task fills or empties an Auto Fill Station (wash station) by pumping for a specified number of seconds or until the percent of maximum tared weight is reached. If the reservoir is on a Weigh Station, the pump stops fluid flow when the target weight is reached. Otherwise, gravity drain is used to empty the reservoir.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

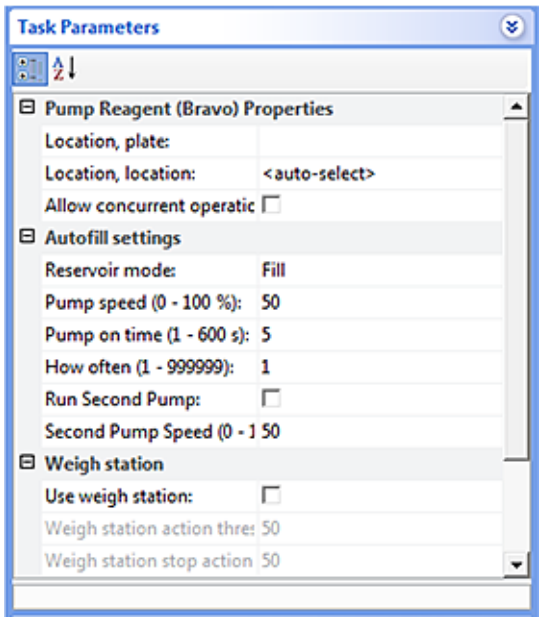
Requirements

The following must be configured on the Bravo Platform:

- Autofilling Reservoir or MicroWash Reservoir (wash station)
- Pump Module
- Weigh Station (optional)

Task parameters

After adding the Pump Reagent task at the desired point in the subprocess, set the following parameters in the **Task Parameters** area:



Parameter	Description
Reservoir mode	The action of the task: • Fill • Empty
Pump speed	The speed, in percent of maximum, at which to pump the reagent.
Pump on time	The duration of time, in seconds, to run the pump: If the Use weigh station option is selected, this value specifies the maximum time (seconds) to run the pump. If the pump runs for the full duration without reaching the Weigh Station threshold, an error occurs.
Use weigh station	The option to use the Weigh Station.
Weigh station action threshold	The minimum fluid weight, in percent of the full weight that was calibrated on the Weigh Station. For example, you can set the minimum threshold at 45% so that when the fluid reaches 45% of the full weight, fluid starts to pump into the reservoir.
Weigh station stop action threshold	The maximum fluid weight, in percent of the full weight that was calibrated on the Weigh Station. For example, you can set the stop threshold at 60% so that when the fluid reaches 60% of the full weight, fluid starts to drain or pump out of the reservoir.

Example: Fill the wash station after the Tip Wash task

Goal

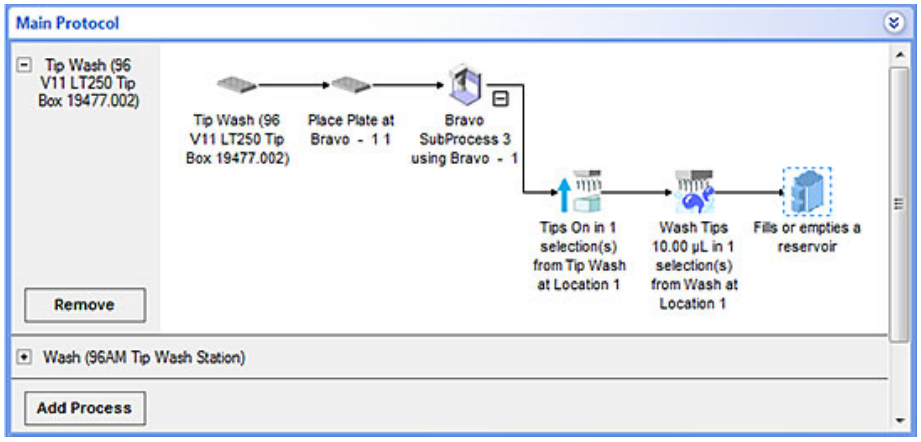
After some liquid-handling tasks, wash the tips, and then fill the wash station.

Implementation

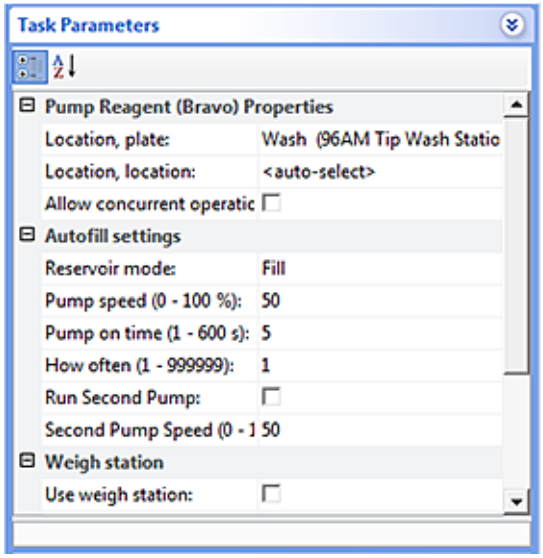
In Bravo Diagnostics, set up an Auto Fill Station (wash station) accessory on a deck location. Associate the wash station with one or more pump modules.

In the protocol, add a configured labware for the wash station in addition to the other processes and configured labware required by the protocol. In the example shown, the configured wash station is called Wash.

In the Bravo subprocess where the liquid-handling tasks are specified, a Pump Reagent task is added after the Wash Tips task. After adding the Pump Reagent task, the task name changes to Fill or empty a reservoir in the protocol, as shown in the following example.



In the Pump Reagents Task Parameters area, Wash (the name for the wash station) is selected, because the goal is to pump fluid into the wash station.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514
Set Head Mode task	"Set Head Mode (Agilent Bravo)" on page 601
Tips On task	"Tips On (Agilent Bravo)" on page 623

For information about...	See...
Tips Off task	"Tips Off (Agilent Bravo)" on page 619
Wash Tips task	"Wash Tips (Agilent Bravo)" on page 632
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Serial Dilution (Agilent Bravo)

Description

The Serial Dilution (Agilent Bravo) () task allows you to set up serial dilution in a microplate using a single task. You use the Serial Dilution wizard to set up the task parameters. The end result is a sequence of Aspirate, Dispense, and optional Mix tasks that produce a linear or non-linear concentration gradient in selected wells.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Requirements

Setup

Make sure you:

- Configure the labware on the Bravo deck in the software. For instructions, see "Configuring labware" on page 53.
- Add a Set Head mode task before the Serial Dilution task to select the pipette channels. See "Set Head Mode (Agilent Bravo)" on page 601 for details.

The Serial Dilution task can be used with the following pipette head and microplate formats:

Series III pipette head	Microplate format
96-channel head with disposable tips	96, 384, 1536
384-channel head with disposable tips	384, 1536

Labware

IMPORTANT

The Serial Dilution task can only be used with a microplate. The task cannot be used with a reservoir.

Make sure the serial dilution microplate meets the following requirements:

- A column, row, or quadrant contains the starting concentration of a compound to be diluted.
- One or more columns, rows, or quadrants contain the same amount of diluent.

Pipette-tip tracking

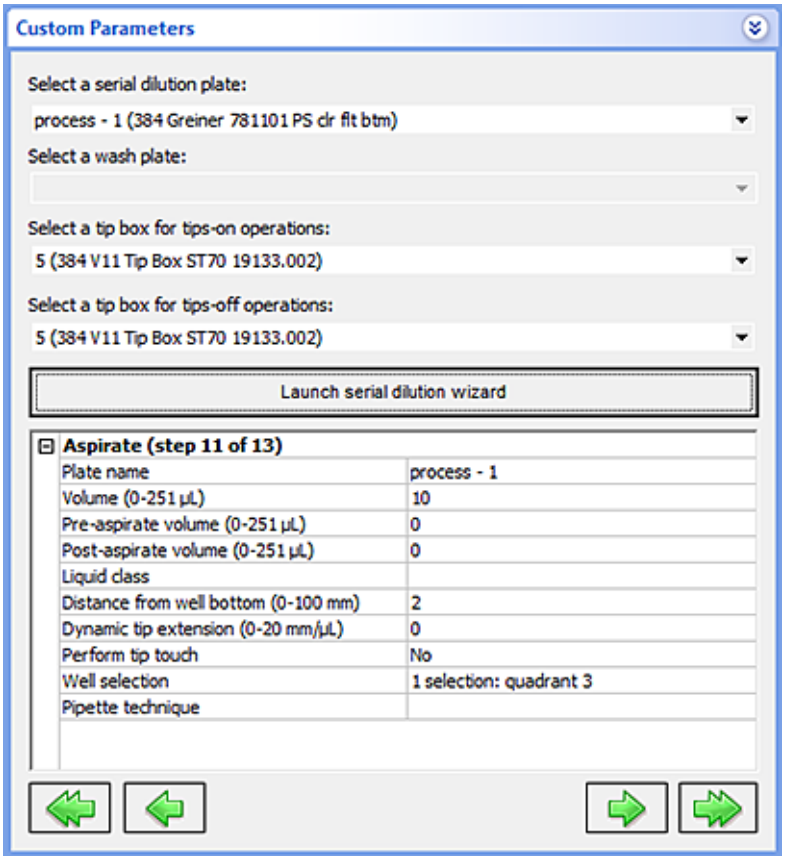
You can track pipette tip usage on the Bravo Platform. To track pipette tip usage, turn on the tracking options in the following:

- Tips On task

- Tips Off task
- Serial Dilution wizard, Step 2

Custom parameters

After adding the Serial Dilution task at the desired point in the protocol, set the following parameters in the **Custom Parameters** area:



Parameter	Description
Select a serial dilution plate	The microplate that will have its contents transferred to other wells. *Note:* The labware must have a base class of Microplate.
Select a wash plate	*Available only if the Tips will be washed option is selected in the wizard*. The reservoir involved in the wash. *Note:* The labware must have a base class of Reservoir.
Select a tip box for tips-on operations	*Available only if the Tips will be changed option is selected in the wizard*. The tip box involved in the Tips On operation.

Parameter	Description
Select a tip box for tips-off operations	*Available only if the Tips will be changed option is selected in the wizard*. The tip box involved in the Tips Off operation.
Launch serial dilution wizard	The command that opens the Serial Dilution Wizard dialog box.

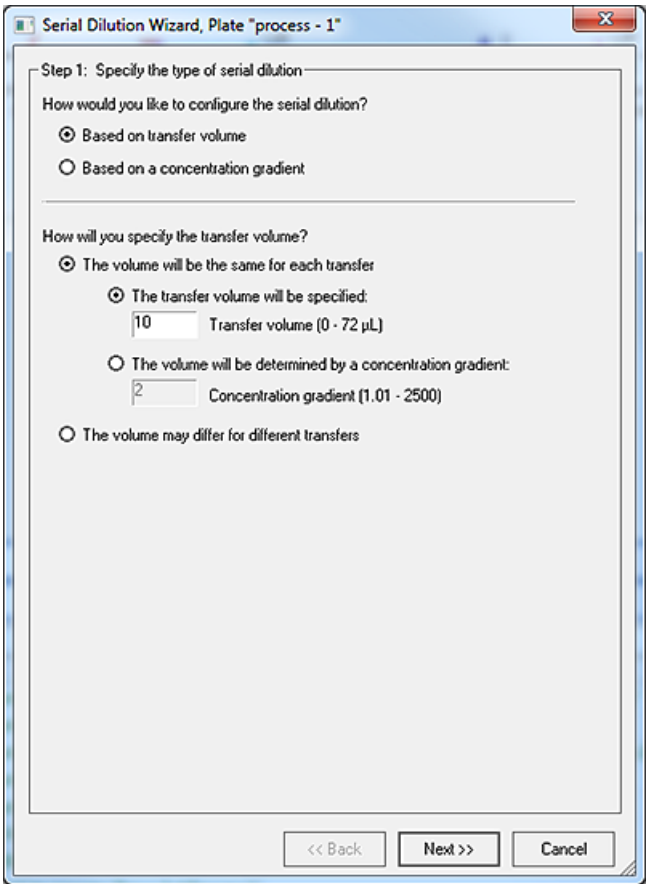
Note: The remaining task parameters will be filled in after you go through the Serial Dilution wizard.

Serial dilution wizard

The serial dilution wizard guides you through the serial dilution setup.

To use the wizard:

1. In the **Task Parameters** area, click **Launch serial dilution wizard**. The Serial Dilution Wizard dialog box opens.
2. Follow the instructions in the wizard to set up the serial dilution.
   a. In Step 1 of the wizard, if you want the transfer volume to be determined by a concentration gradient, be sure to type a gradient factor.

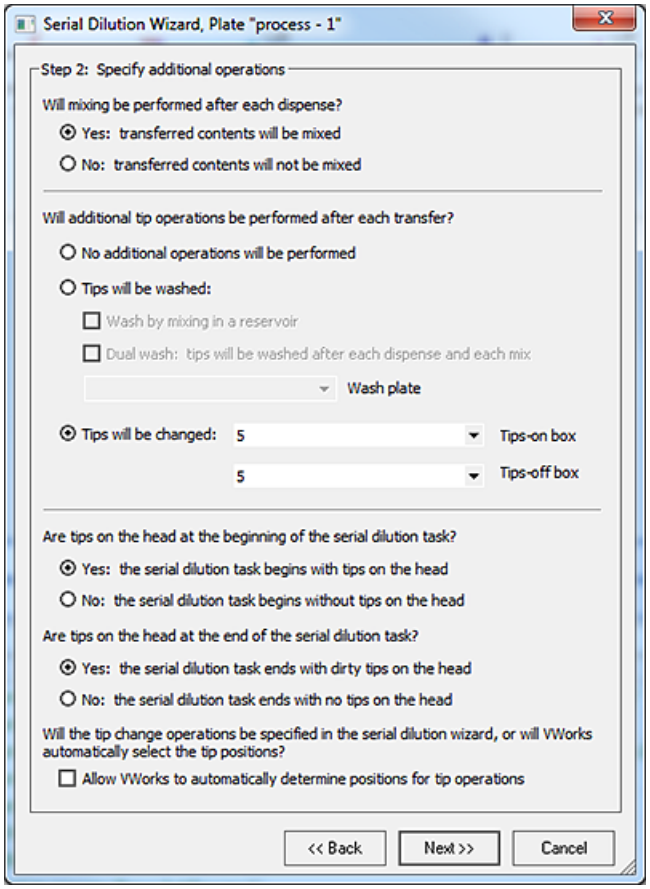


For example, if the concentration gradient is 2, then the concentration of the first dilution will be the concentration in the starting column C_1 divided by 2, or $C_1/2$. The concentration of the second dilution will be the concentration of the second column C_2 divided by 2, or $C_2/2$, and so on.

The upper range of the transfer volume is determined by the capacity of the pipette head and the well volume of the microplate.

Note: If you select Based on a transfer volume, **The volume will be determined by a concentration gradient** option is not available.

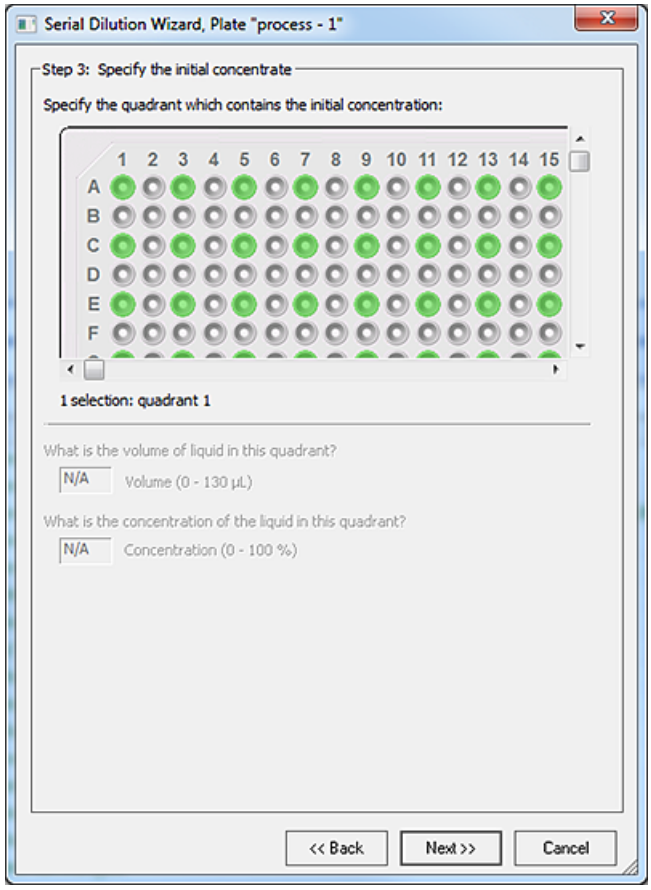
b In Step 2 of the wizard, select whether you want to mix the contents, wash the tips, or have the software track tip usage.



IMPORTANT If you want to track tips, make sure you also select the tracking options in the Tips On and Tips Off tasks. For instructions, see "Tips On (Agilent Bravo)" on page 623 and "Tips Off (Agilent Bravo)" on page 619.

c In Step 3, specify the initial concentrate.

Note: The Volume and Concentration controls are available only if **Based on a concentration gradient** was selected in Step 1 of the wizard.



d Follow the instructions in Step 4 and Step 5 of the wizard to specify the initial concentrate and verify the transfers.

Note: In Step 5, the **Transfer volume** is available only if **The volume may differ for different transfers** was selected in Step 1 of the wizard. If the desired concentration in the target wells cannot be reached using the specified value, it will appear in red.

e Follow the instructions in Step 6 to select the Aspirate and Dispense parameters.

- For a description of the Aspirate parameters, see "Aspirate (Agilent Bravo)" on page 455.
- For the description of the Dispense parameters, see "Dispense (Agilent Bravo)" on page 514.

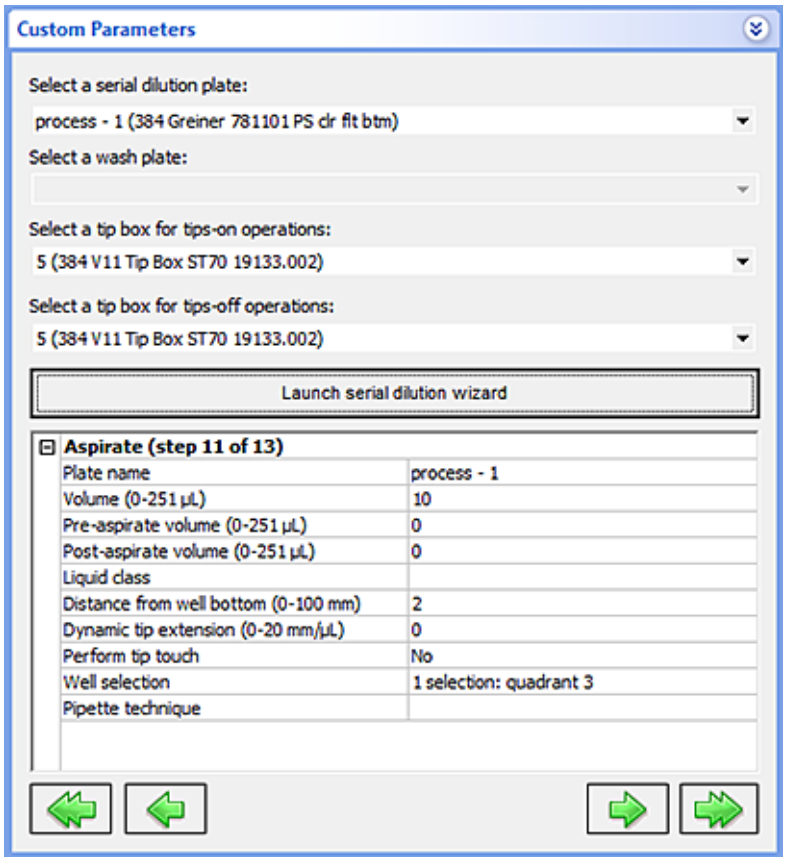
f Follow the instructions in Step 7 to select the Mix parameters. For the description of the Mix parameters, see "Mix (Agilent Bravo)" on page 559.

g In Step 8 of the wizard:

- If you selected **Tips will be changed** in Step 2, specify the tip changes.
- If you selected **Tips will be washed** in Step 2, specify the wash parameters. For the description of the wash parameters, see "Wash Tips (Agilent Bravo)" on page 632 or "AM Wash Syringes (Agilent Bravo)" on page 494, as applicable.

3 Click **Finish** when you are finished setting up the serial dilution task.

4 In the **Custom Parameters** area, review the parameters you specified in the wizard. To see additional parameters, click the arrow buttons under the parameter table. If you need to edit a parameter value, double-click in the box and type the new value.



Related information

For information about...	See...
Configured labware	"Configuring labware" on page 53
Static labware	"Planning labware use" on page 24
Startup Protocol	"Setting up Startup and Cleanup Protocol processes" on page 75
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Set Head Mode (Agilent Bravo)

Description

The Set Head Mode (Agilent Bravo) ( Set Head Mode (Agilent Bravo)) task specifies the channels (or barrels) in the pipette head to be used for pipetting. You can select an m × n array of channels (barrels) for one of the following configurations:

- All of the pipette channels
- The first or last full column or row of pipette channels
- Multiple full columns or rows of pipette channels
- The first or last partial column or row of pipette channels
- Multiple partial columns or rows of pipette channels
- A single pipette barrel at the corner of the pipette head

This task should only be used if the Series III pipette head is installed.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

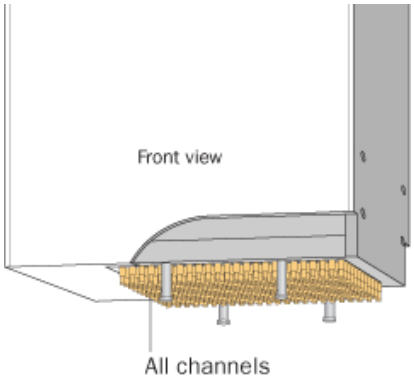
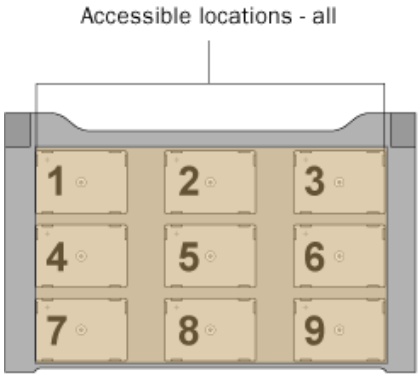
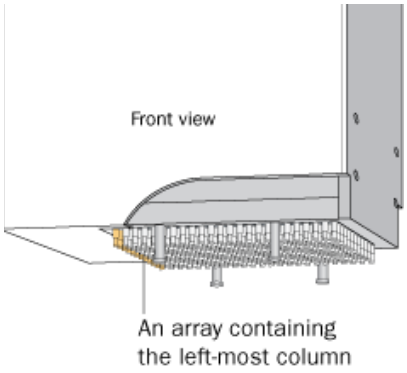
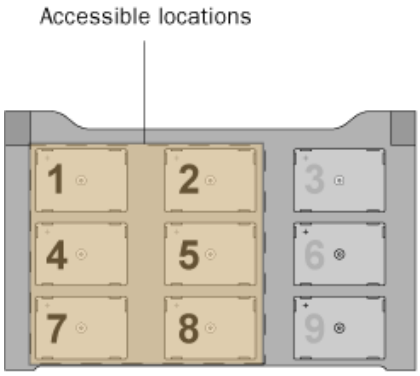
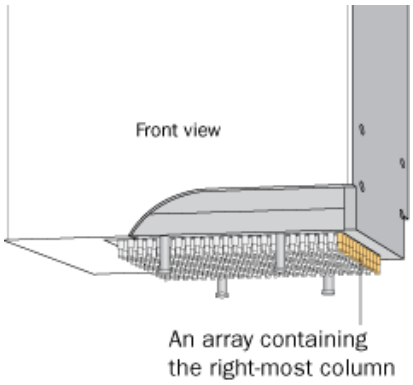
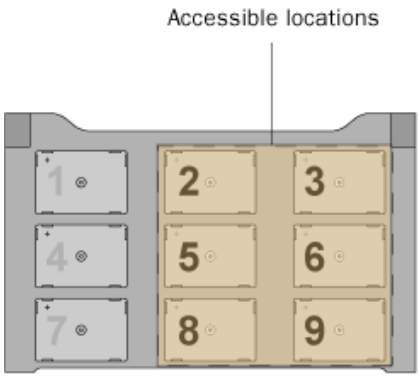
Liquid-handling head requirements

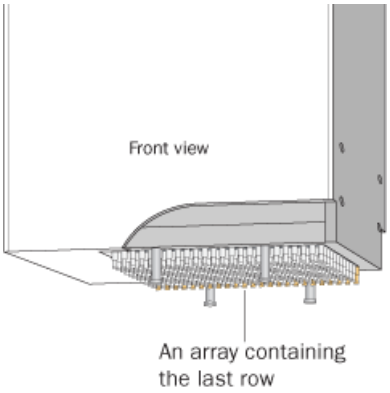
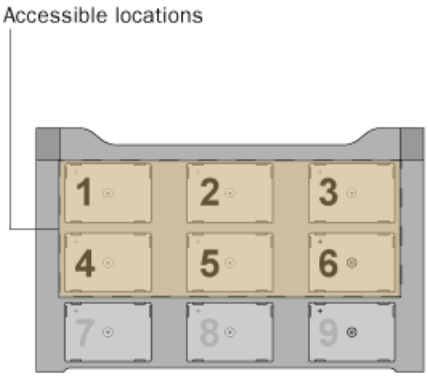
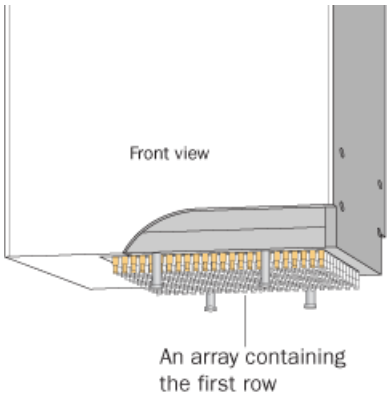
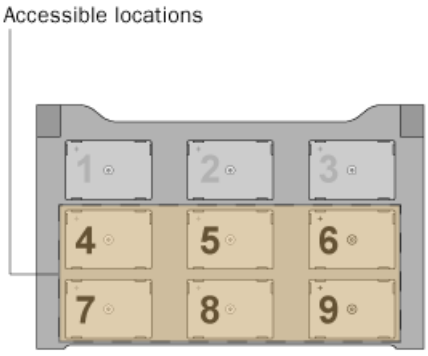
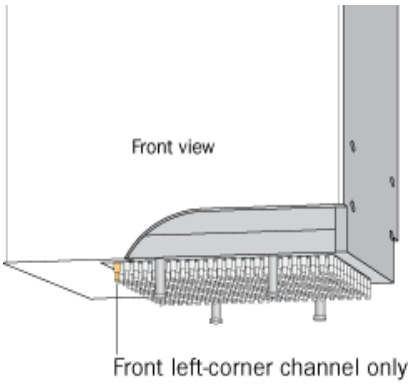
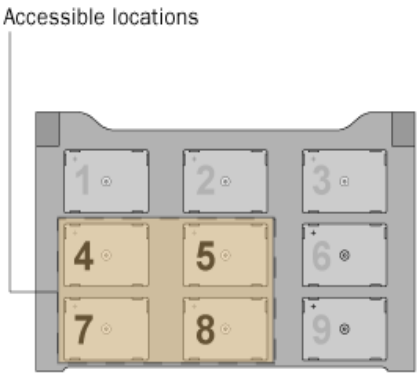
You can use the following pipette heads with a flexible array of pipette tips:

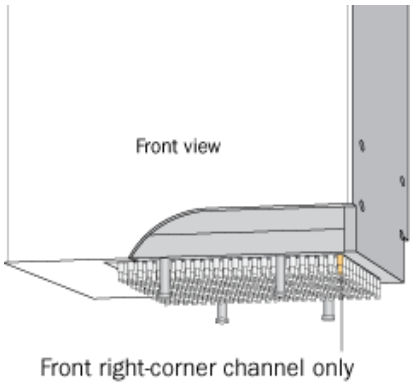
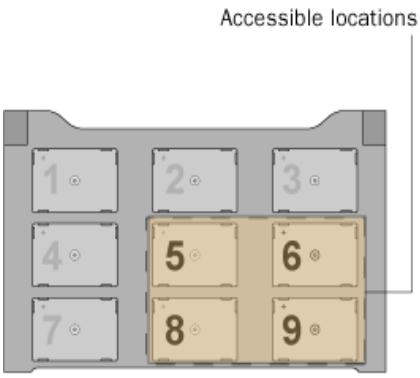
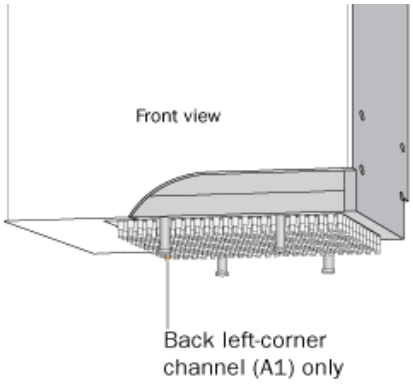
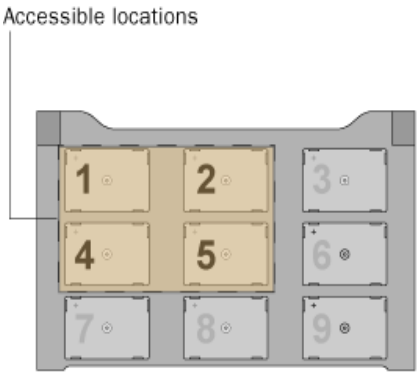
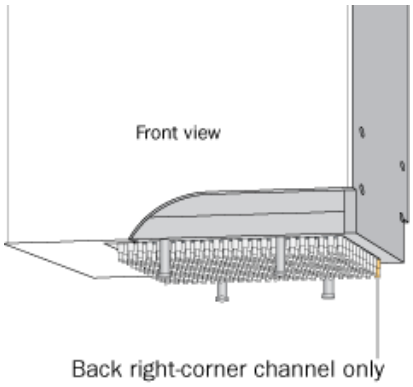
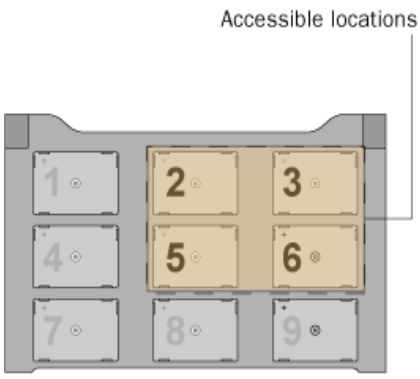
Pipette head	Microplate format
Series III 96-channel head with disposable tips	96, 384, or 1536
Series III 384-channel head with disposable tips	96, 384 or 1536

Accessible deck locations

The deck locations you can access depends on the pipette head channels you select. The following table shows the channel selections and corresponding deck location limits. Use this information when you set up labware on the Bravo deck.

Pipette channel selection	Accessible Bravo deck locations
	
	
	

Pipette channel selection	Accessible Bravo deck locations
	
	
	

Pipette channel selection	Accessible Bravo deck locations
	
	
	

Before you add the task

Make sure:

- You have installed the correct pipette head. See "Liquid-handling head requirements" on page 601 and the *Bravo Platform User Guide*.
- If you are going to use partial rows or columns of channels (barrels) on the pipette head, retract the tip box stripper pins on the pipette head. See the *Bravo Platform User Guide* for this procedure.
- The correct Bravo device file is open in the VWorks window.

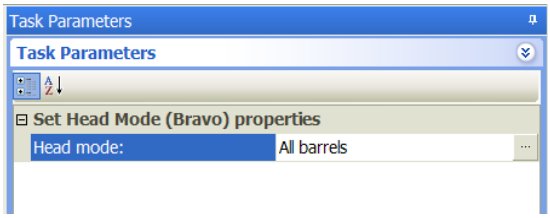
- The profile you selected shows the correct head type, tip type, and miscellaneous settings.
- All the teachpoints have been added and verified.

Task parameters

IMPORTANT The Set Head Mode task should always precede the pipetting tasks that require the specified subset of pipettes.

IMPORTANT The Set Head Mode task should precede the Serial Dilution task. If you plan to change tips during the serial dilution process, add the Set Head Mode task before the Tips On task.

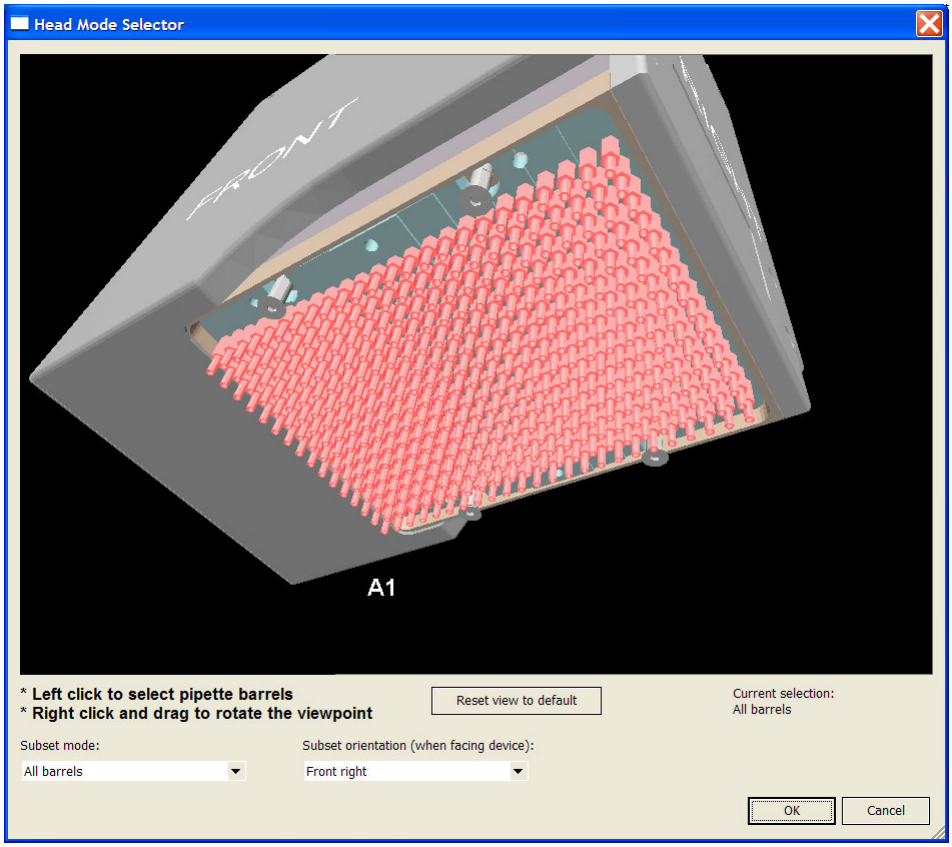
After adding the Set Head Mode task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Head mode	The channel selection. Click the Head mode field, and then click the ... button that appears. In the Head Mode Selector dialog box that opens, select the pipette channels.

Head Mode Selector dialog box

The Head Mode Selector dialog box allows you to select the pipette channels. Except for the **Subset: All barrels** mode, you select the desired channels using a combination of the **Subset mode** selection with the **Subset orientation** selection.



To select the pipette channels:

1 In the **Subset** list, select one of the following:

Subset mode	Description
All barrels	Uses all of the pipette channels.
Full column	Uses one or more full columns of pipette channels, starting from the right-most or left-most column.
Full row	Uses one or more full rows of pipette channels, starting from the first row or the last row.
Partial row/column	Uses part of the selected columns or rows.

2 In the **Subset orientation** list, select one of the following:

Subset orientation	Description
Front right	Uses pipette channels that contain the single channel in the front right corner.
Back right	Uses pipette channels that contain the single channel in the back right corner.
Front left	Uses one or more full rows of pipette channels, starting from the first row or the last row.
Back right	Uses part of the selected columns or rows.

3 When you are finished, click **OK** to save the selection.

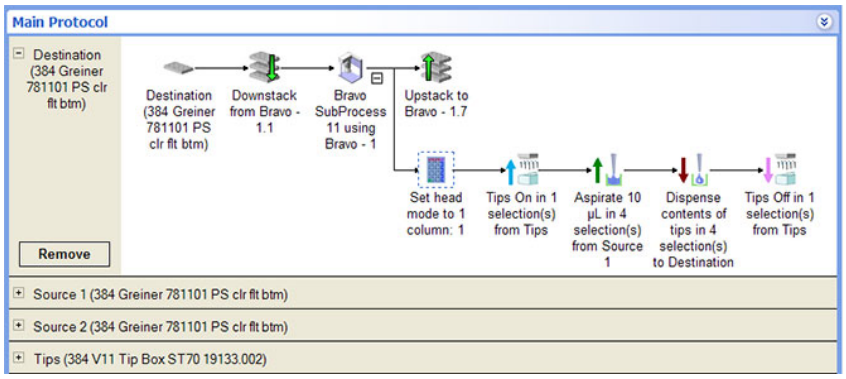
Example: Specify the pipette channels to use on the Bravo Platform

Goal

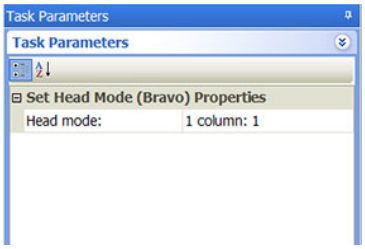
Use only column 1 of a Series III 96-channel pipette head for the liquid-handling tasks.

Implementation

At the beginning of the Bravo subprocess, add a Set Head Mode task as shown.



In the Set Head Mode Task Parameters area, column 1 is selected.



Related information

For information about...	See...
Configured labware	"Configuring labware" on page 53
Static labware	"Planning labware use" on page 24
Startup Protocol	"Setting up Startup and Cleanup Protocol processes" on page 75
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Tips On task	"Tips On (Agilent Bravo)" on page 623
Tips Off task	"Tips Off (Agilent Bravo)" on page 619
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

For information about...	See...
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Set Temperature (Agilent Bravo)

Description

The Set Temperature (Agilent Bravo) ( Set temperature (Bravo)) task sets or turns off the target temperature for a Heating Shaking Station accessory while a specific labware is on the station.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

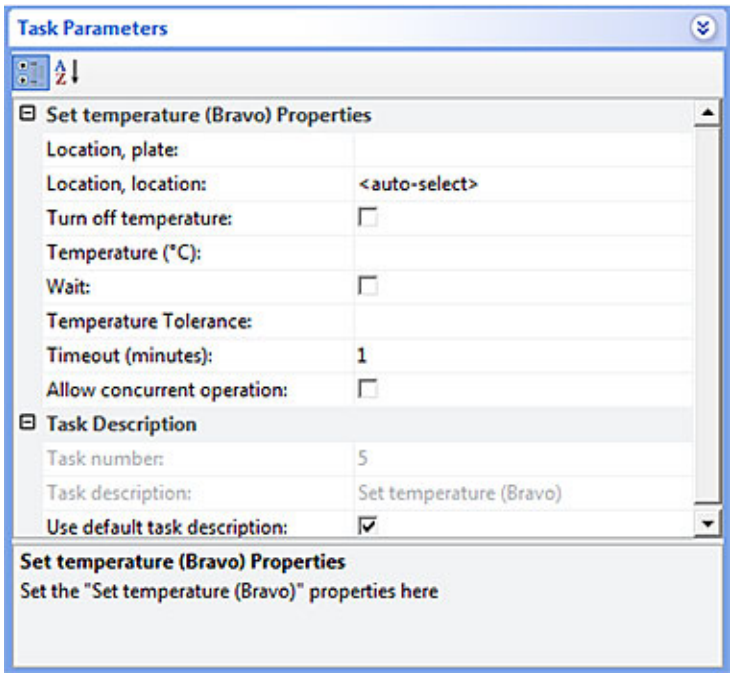
If you want to control the target temperature for a Heating Shaking Station accessory without regard to which labware is on the station, see "Set Temperature - Position (Agilent Bravo)" on page 611.

Requirements

To use the Set Temperature task, you must first configure a Heating Shaking Station accessory in Bravo Diagnostics. The software considers any heating or shaking accessory controlled by an Inheco STC or MTC Controller to be a Heating Shaking Station accessory, for example, the Peltier Thermal Station (CPAC).

Task parameters

After adding the Set Temperature task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The process or static labware involved in the task.
Location, location	The deck location of the Heating Shaking Station accessory. If more than one Heating Shaking Station accessory is on the deck, select the specific deck location instead of <auto-select>. *Note:* Temperature control will not commence or terminate until the labware is in place.
Turn off temperature	The option to use the Set Temperature task to turn off the temperature control. If this option is not selected, the task will start temperature control.
Temperature	The target temperature (°C) for the accessory to achieve during this task.
Wait	The option to wait until the accessory achieves the target temperature within the specified tolerance before the task completes. Subsequent tasks in the same subprocess cannot execute until this task completes. If this option is not selected, the software considers the task completed as soon as the temperature control turns on or off successfully.
Temperature tolerance	The acceptable range (°C) above and below the specified target temperature still considered to be the target temperature.
Timeout	The duration (minutes) to wait for the temperature to reach the specified target temperature within the specified tolerance.
Allow concurrent operation	Not applicable

Related information

For information about...	See...
Configured labware	"Configuring labware" on page 53
Static labware	"Planning labware use" on page 24
Startup Protocol	"Setting up Startup and Cleanup Protocol processes" on page 75
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67

For information about...	See...
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Set Temperature - Position (Agilent Bravo)

Description

The Set Temperature - Position (Agilent Bravo) () task sets or turns off the target temperature for a Heating Shaking Station accessory, without regard to which labware is on the station.

Task is available for...	Task is available in...
Bravo Platform	Startup Protocol Main Protocol, Bravo Subprocess Cleanup Protocol

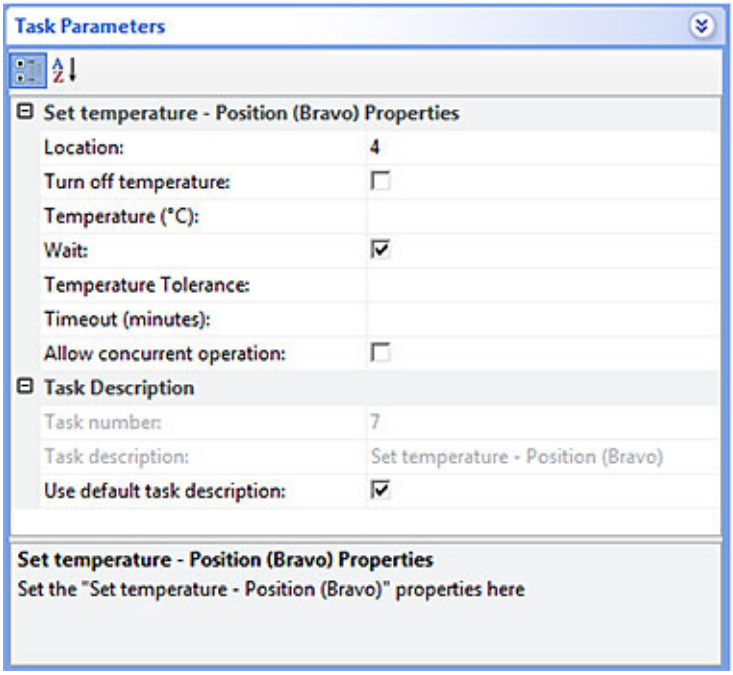
If you want to control the temperature for a Heating Shaking Station accessory for a specific labware on the station, see "Set Temperature (Agilent Bravo)" on page 609.

Requirements

To use the Set Temperature - Position task, you must first configure a Heating Shaking Station accessory in Bravo Diagnostics. The software considers any heating or shaking accessory controlled by an Inheco STC or MTC Controller to be a Heating Shaking Station accessory, for example, the Peltier Thermal Station (CPAC).

Task parameters

After adding the Set Temperature - Position task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location	The Bravo deck location where the task will occur, for example, a Peltier Thermal Station (CPAC) at deck location 4 or 6. If the task is in a Bravo subprocess, the Bravo device specified by the Subprocess task is used. If the task is in the Startup or Cleanup protocol, the Task Parameters pane Device Selection specifies which Bravo to use.
Turn off temperature	The option to use this task to turn off the temperature control. If this option is not selected, the task will start temperature control.
Temperature	The target temperature (°C) for the accessory to achieve during this task.
Wait	The option to wait until the accessory achieves the target temperature within the specified tolerance before the task completes. If this option is not selected, the task considers itself completed as soon as the temperature control turns on or off successfully.
Temperature tolerance	The acceptable range of temperature (°C) above or below the specified target temperature.
Timeout	The duration (minutes) to wait for the temperature to reach the specified target temperature within the specified tolerance.

Parameter	Description
Allow concurrent operation	Not applicable

Related information

For information about...	See...
Configured labware	"Configuring labware" on page 53
Static labware	"Planning labware use" on page 24
Startup Protocol	"Setting up Startup and Cleanup Protocol processes" on page 75
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Shake (Agilent Bravo)

Description

The Shake (Agilent Bravo) ( Shake (Bravo)) task instructs the Orbital Shaking Station to shake.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

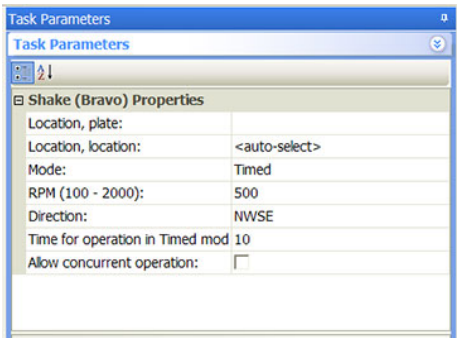
Requirements

To use the Shake task, you must first configure the Orbital Shaking Station in Bravo Diagnostics.

If you are using an Orbital Shaking Station that is connected to an Inheco STC or MTC Controller, use the Shake Plate task instead. See "Shake Plate (Agilent Bravo)" on page 616.

Task parameters

After adding the Shake task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The labware involved in the Shake task.
Location, location	The location at which the Shake task occurs. <auto-select> automatically places the labware at the first-available or appropriate location for the task. If accessories are installed on the deck, the software uses the accessory configuration information in Bravo Diagnostics to determine the correct location for the task.
Mode	The action of the task: • *On.* Turns on the Orbital Shaking Station. • *Off.* Turns off the Orbital Shaking Station. • *Timed.* Turns on the shaking timer. You must specify the length of time to shake. If you plan to time the shaking, add only one Shake task (the task turns on the shaking, and at the end of the time period, the shaking turns off automatically). If you are not timing the shaking, add two Shake tasks in the protocol (one to turn on the shaking and the other to turn off the shaking).
RPM	The shake speed, in revolutions per minute.
Direction	The direction to shake. Select one of the direction combinations: NWSE, NESW, NS, EW, NW/SE, NE/SW.
Time for operation in Timed mode	The length of time, in seconds, you want to leave the shaking on. At the end of the period, the shaking will turn off.

Parameter	Description
Allow concurrent operation	The option to permit the accessory to operate simultaneously with other tasks. For example, the Shake and Mix tasks can operate simultaneously. **CAUTION** To shake and mix concurrently, use only 96-well disposable-tip pipette heads in 96-well microplates.

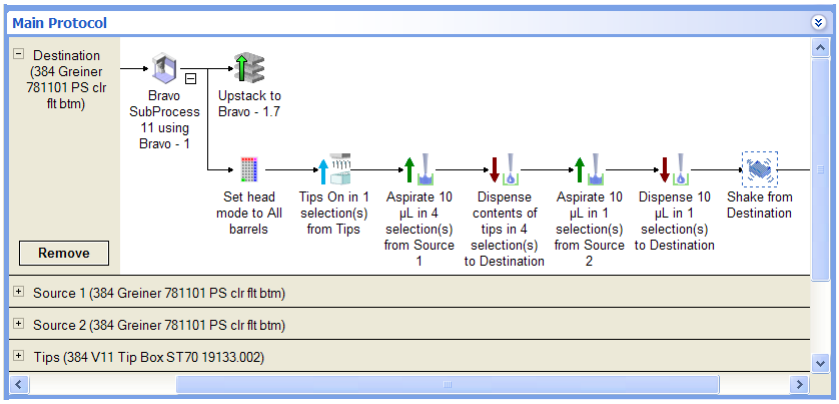
Example: Shake the destination microplate after adding reagents

Goal

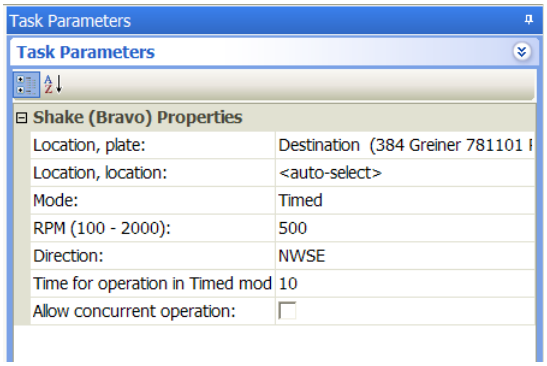
Shake the destination microplate after adding reagents from two source microplates.

Implementation

In the liquid-handling subprocess, a Shake task is added after the second reagent is added. In the following example, a Shake task is added in a Bravo subprocess.



In the Shake Task Parameters area, Destination is selected, because the goal is to shake the destination microplate.



Related information

For information about...	See...
Configured labware	"Configuring labware" on page 53

For information about...	See...
Static labware	"Planning labware use" on page 24
Startup Protocol	"Setting up Startup and Cleanup Protocol processes" on page 75
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Shake Plate (Agilent Bravo)

Description

The Shake Plate (Agilent Bravo) ( Shake plate (Bravo)) task instructs a Heating Shaking Station accessory to shake.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

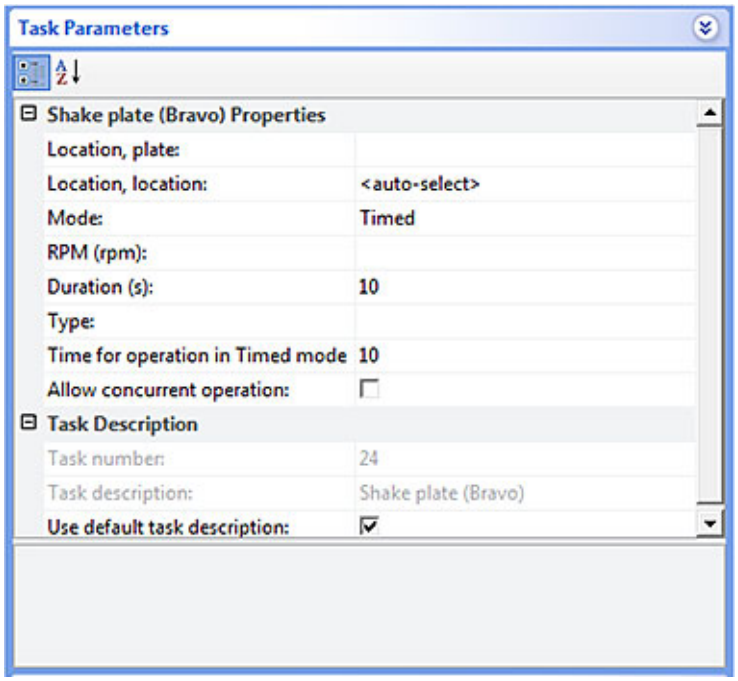
Requirements

To use the Shake Plate task, you must first configure a Heating Shaking Station accessory in Bravo Diagnostics. The software considers any heating or shaking accessory controlled by an Inheco STC or MTC Controller to be a Heating Shaking Station accessory, for example, the Orbital Shaking Station or a Heating Shaking Station.

If you are using an Orbital Shaking Station that has its own control module, use the Shake task instead. See "Shake (Agilent Bravo)" on page 613.

Task parameters

After adding the Shake Plate task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The process or static labware involved in the task.
Location, location	The deck location of the Heating Shaking Station accessory. If more than one Heating Shaking Station accessory is on the deck, select the specific deck location instead of <auto-select>.
Mode	The action of the task: • *On.* Turns on the Heating Shaking Station accessory. • *Off.* Turns off the Heating Shaking Station accessory. • *Timed.* Turns on the shaking timer. You must specify the length of time to shake. If you plan to time the shaking, add only one Shake Plate task (the task turns on the shaking, and at the end of the timed period, the shaking turns off automatically). If you are not timing the shaking, add two Shake Plate tasks in the protocol (one to turn on the shaking and the other to turn off the shaking).
RPM	The shake speed, in revolutions per minute.
Duration	*Timed Mode only*. The length of time, in seconds, to leave the shaking on. At the end of the period, the shaking will turn off.

Parameter	Description
Type	The direction to shake. The choices are Circle Anticlockwise, Circle Clockwise, Up Left Down Right, Up Right Down Left, Up Down, and Left Right.
Time for operation in Timed mode	The value that the software uses only for validating the task during protocol compilation. When running the protocol, the Duration value is used instead.
Allow concurrent operation	*Timed Mode only*. The option to permit the accessory to operate simultaneously with other tasks. For example, the Shake and Mix tasks can operate simultaneously. **CAUTION** To shake and mix concurrently, use only 96-well disposable-tip pipette heads in 96-well microplates.

Related information

For information about...	See...
Configured labware	"Configuring labware" on page 53
Static labware	"Planning labware use" on page 24
Startup Protocol	"Setting up Startup and Cleanup Protocol processes" on page 75
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Tips Off (Agilent Bravo)

Description

The Tips Off (Agilent Bravo) () task ejects disposable pipette tips from the liquid-handling heads that are compatible with disposable tips.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Options in the Tips On and Tips Off tasks permit tracking of pipette tip usage during and across protocol runs. In addition, an option in the Tips Off task allows you to reuse pipette tips for a portion of the protocol run. If the Bravo Platform is configured with a trash accessory, you can discard of used pipette tips at the trash location.

Requirements

Tracking tips in serial dilution tasks

In addition to selecting the tracking option in the Tips On and Tips Off tasks, you must also turn on the tracking in the Serial Dilution Wizard. For more detailed information, see "Serial Dilution (Agilent Bravo)" on page 595.

Tracking tips across different protocols

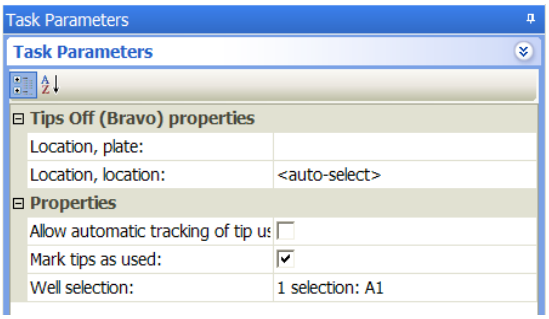
If you want to track pipette tip usage across different protocols that contain Bravo Subprocesses, make sure:

- The protocols reference the same device file.
- The tip boxes are at the same physical locations.
- In the software, the tip boxes are configured at the same locations across the protocols.

Note: When a set of tip boxes are designated as a process plate, tip usage is tracked during the protocol run. At the end of a run, the software resets the tip box to the original state. The software assumes that in each subsequent run, you will load tip boxes in the original state before the run.

Task parameters

After adding the Tips Off task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The labware involved in the Tips Off task.
Location, location	The location at which the Tips Off task occurs. <auto-select> automatically places the labware at the first-available or appropriate location for the task. If accessories are installed on the deck, the software uses the accessory configuration information in Bravo Diagnostics to determine the correct location for the task.
Allow automatic tracking of tip usage	The option to allow the software to track pipette tip usage during the protocol run or across different protocol runs. When you start the run, the software will determine the positions to use in the tip box. If you select the option in this task, you must also select the option in the Tips On task. If you do not select this option, you must specify the positions to use in the tip box using the Well selection parameter.
Mark tips as used	The option to use only new pipette tips during the protocol run. Select the option so that the software counts the number of tips used during the protocol run. The tips that have been used once are marked as used so that they cannot be picked up and reused. Clear the check box so that during the next Tips On task, the same tips can be reused.
Well selection	The well positions to use for the Tips Off task. This parameter is available only for manual tracking of pipette tips.

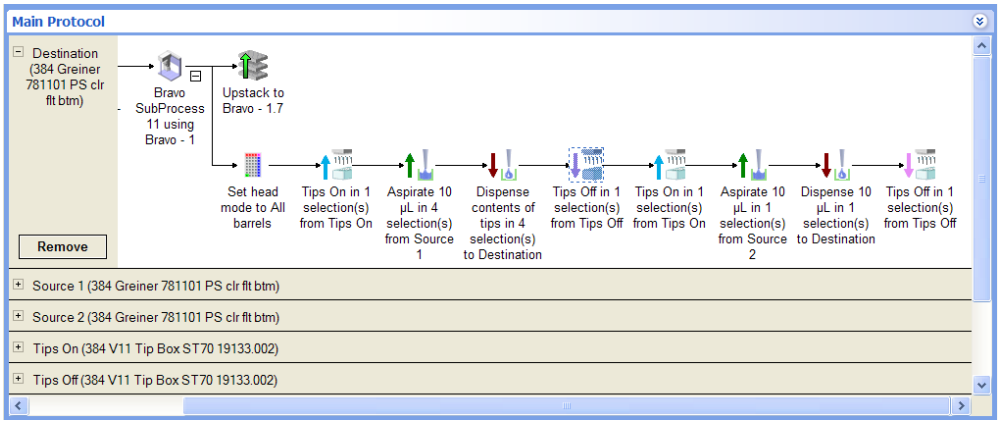
Example: Change tips during Bravo liquid-handling tasks

Goal

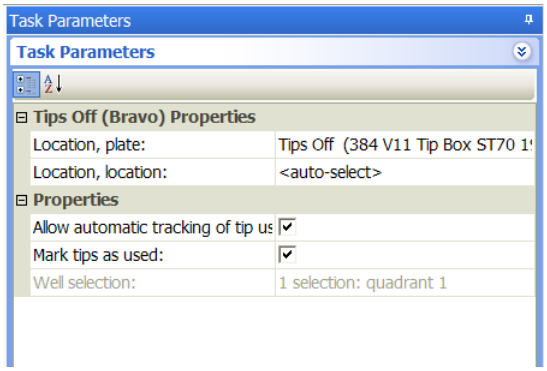
Add reagents from two source microplates into a destination microplate. Change tips between source microplate 1 and source microplate 2 to prevent contaminating the source microplates.

Implementation

In the liquid-handling subprocess, add a Tips Off task after each Dispense task as shown.



Because the tips are removed at a Tips Off tip box, the Tips Off tip box is selected in the Tips Off Task Parameters area.



Example: Discard pipette tips at Bravo trash location

Goal

Dispose of used pipette tips in a trash container.

Implementation

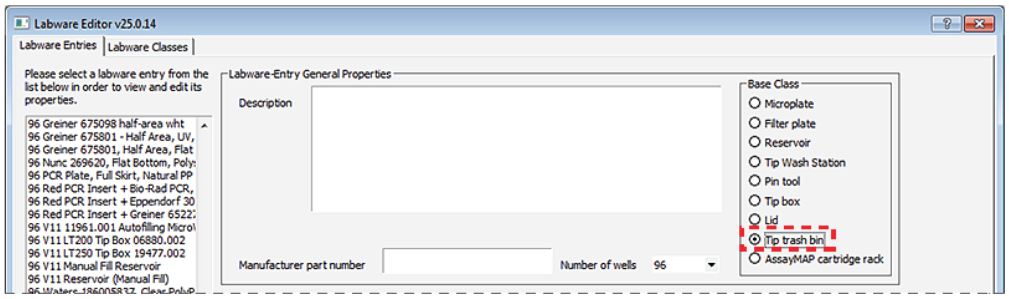
1 In the device file, configure the Trash or Filter Plate Holder accessory in the Bravo profile. For details, see the *Bravo Platform User Guide*.

IMPORTANT

The Trash or Filter Plate Holder accessory can be installed at Bravo deck location 4 or 6 only.

2 Ensure that the Labware Editor includes an entry for Tip Trash that is associated with the Base Class: Tip trash bin. For details on the Labware Editor, see the *VWorks Automation Control Setup Guide*.

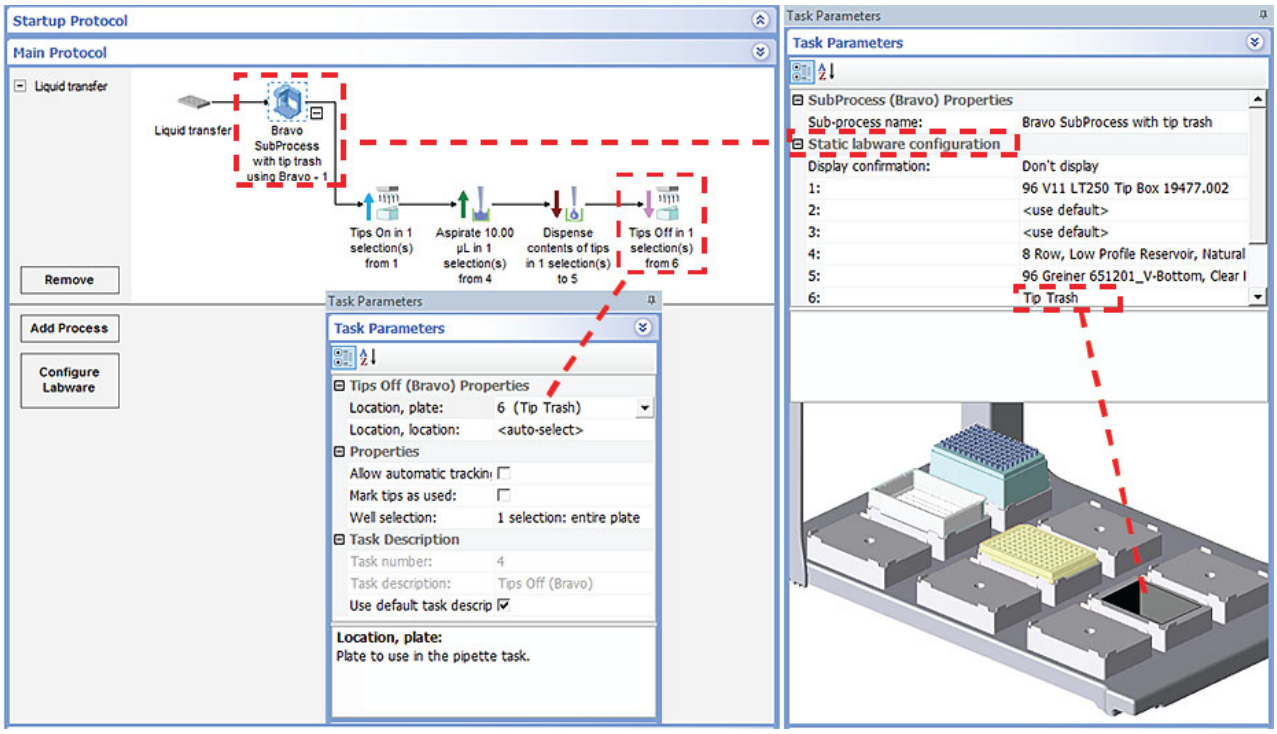
Figure Tip trash bin base class option in the Labware Editor dialog box



3 In the protocol, ensure the Bravo SubProcess includes a static labware configuration for Tip Trash. For details on configuring labware, see "Configuring labware" on page 53.

4 To remove the tips at the Bravo trash location, ensure that Tip Trash is selected in the Tips Off Task Parameters area.

The following figure shows a simple protocol where all the labware in the subprocess, as well as the Tip Trash are configured as static labware.



Related information

For information about...	See...
Tips On task	"Tips On (Agilent Bravo)" on page 623

For information about...	See...
Serial Dilution task	"Serial Dilution (Agilent Bravo)" on page 595
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639
Setting up the Bravo Platform	*Bravo Platform User Guide*

Tips On (Agilent Bravo)

Description

The Tips On (Agilent Bravo) (

) task presses disposable pipette tips onto the liquid-handling head. The task is not for use with fixed-tip pipette heads.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Note: Options in Tips On and Tips Off tasks permit the tracking of pipette tip usage during and across protocol runs. In addition, an option in the Tips Off task allows you to reuse pipette tips for a portion of the protocol run.

Requirements

Tracking tips in serial dilution tasks

In addition to selecting the tracking option in the Tips On and Tips Off tasks, you must also turn on the tracking in the Serial Dilution Wizard. For more detailed information, see "Serial Dilution (Agilent Bravo)" on page 595.

Tracking tips across different protocols

If you want to track pipette tip usage across different protocols that contain Bravo Subprocesses, make sure:

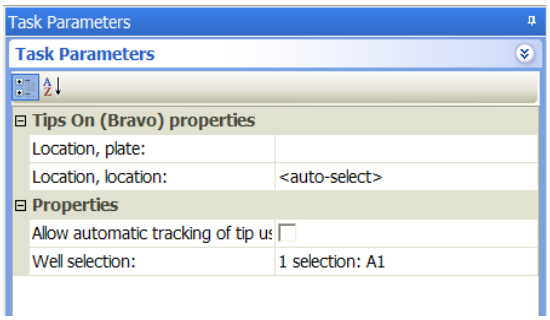
- The protocols reference the same device file.
- The tip boxes are at the same physical deck locations.

- In the software, the tip boxes are configured at the same locations across the protocols.

Note: When a set of tip boxes are designated as a process plate, tip usage is tracked during the protocol run. At the end of a run, the software resets the tip box to the original state. The software assumes that in each subsequent run, you will load tip boxes in the original state before the run.

Task parameters

After adding the Tips On task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location, plate	The labware involved in the Tips On task.
Location, location	The location at which the Tips On task occurs. <auto-select> automatically places the labware at the first-available or appropriate location for the task. If accessories are installed on the deck, the software uses the accessory configuration information in Bravo Diagnostics to determine the correct location for the task.
Allow automatic tracking of tip usage	The option to allow the software to track pipette tip usage during the protocol run or across different protocol runs. When you start the run, the software will determine the positions to use in the tip box. If you select the option in this task, you must also select the option in the Tips Off task. If you do not select this option, you must specify the positions to use in the tip box using the Well selection parameter.
Well selection	The well positions to use for the Tips On task. This parameter is available only for manual tracking of pipette tips.

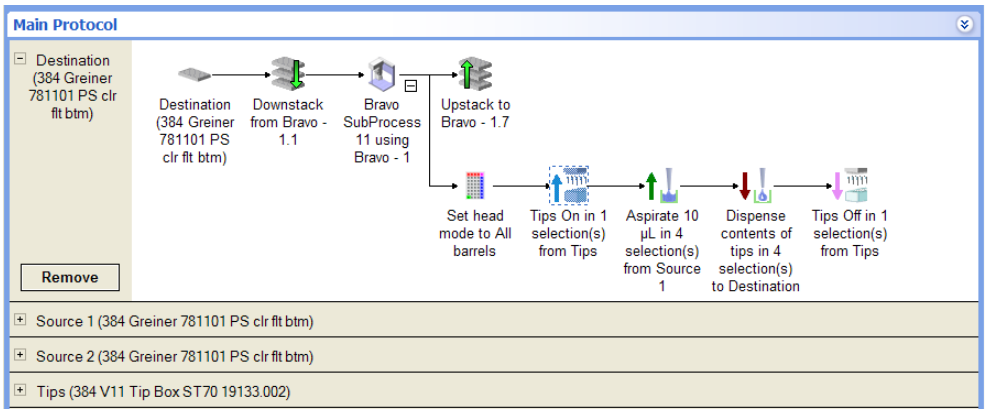
Example: Press tips on before Bravo liquid-handling tasks

Goal

Make sure tips are pressed on before liquid-handling tasks start.

Implementation

In the liquid-handling subprocess, a Tips On task is added at the beginning. In the following example, because the protocol is run on a Bravo Platform, a Set Head Mode task must be added before the Tips On task.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Tips On task	"Tips On (Agilent Bravo)" on page 623
Serial Dilution task	"Serial Dilution (Agilent Bravo)" on page 595
Set Head Mode task	"Set Head Mode (Agilent Bravo)" on page 601
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Toggle Vacuum (Agilent Bravo)

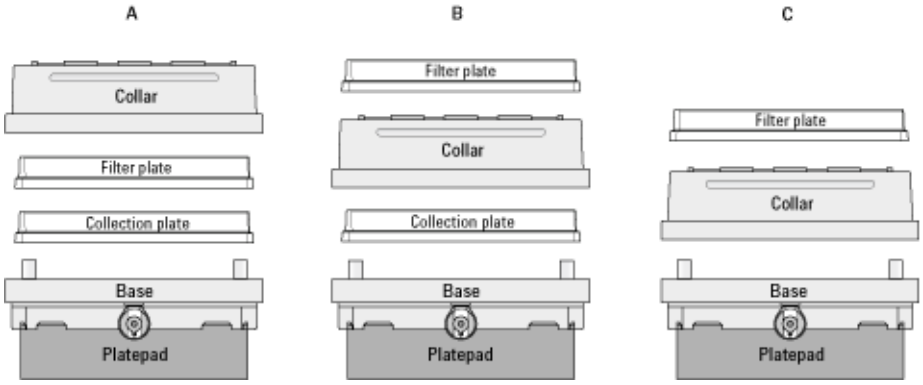
Description

The Toggle Vacuum (Agilent Bravo) (Toggle Vacuum (Bravo)) task turns on and turns off the vacuum in a Vacuum Filtration Station.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Use the Toggle Vacuum (Bravo) task if the Vacuum Filtration Station has configuration A as shown in the following figure. In configuration A, the filter plate is part of the station assembly.

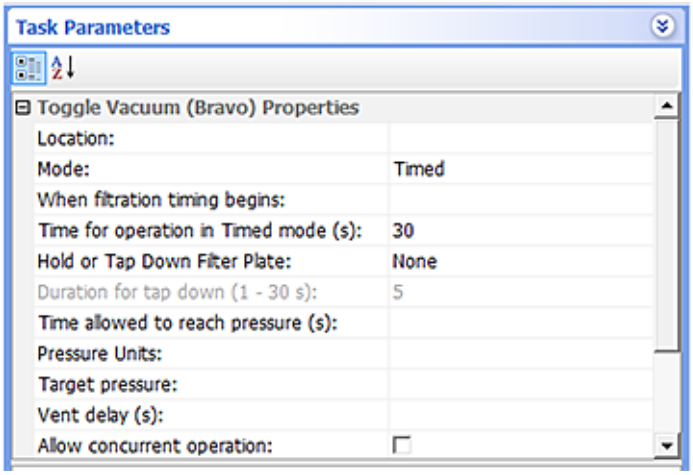
Figure Vacuum Filtration Station configurations A, B, and C



If the Vacuum Filtration Station has configuration B or C and you want the robot to move the filter plate from another location to the station, use the Move and Filter Plate task to turn on and turn off the vacuum. See "Move and Filter Plate (Agilent Bravo)" on page 568.

Task parameters

After adding the Toggle Vacuum task at the desired point in the subprocess, set the following parameters in the **Task Parameters** area:



Parameter	Description
Location	The location of the Vacuum Filtration Station.
Mode	The action of the task. The options are On, Off, and Timed. • **On and Off**. If you are not timing the filtering process, add two Toggle Vacuum tasks in the protocol for each filtering operation. One task turns on the vacuum (Mode = On), and the other task turns off the vacuum (Mode = Off). • **Timed**. If you plan to time the filtering process, add only one Toggle Vacuum task. The task turns on the vacuum, and then turns off the vacuum automatically at the end of the time period.
When filtration timing begins *ME4C VARIO Vacuum Pump only*	The different options for when to start timing the filtration process: • When pressure is achieved • When pump starts
Time for operation in Timed mode	The length of time, in seconds, you want to leave the vacuum on. At the end of the period, the vacuum will turn off.

Parameter	Description
Hold or tap down filter plate	The different options for whether to have the Bravo gripper hold down the filter plate when the vacuum is turned on to ensure a secure vacuum seal: • **None**. The Bravo gripper will not hold down the filter plate. • **Tap down**. The Bravo gripper will hold down the filter plate from the top only for the time period specified in the Duration for tap down parameter. • **Hold down**. The Bravo gripper will hold down the filter plate from the top for the duration of the task. *Note:* If the Vacuum Filtration Station is assembled in configuration A without a filter plate on top of the collar, and the Tap down or Hold down option is selected, the gripper will press directly on the outer upper plastic rim of the collar. *Note:* If you select None or Tap down, the protocol can perform other tasks in parallel. Concurrent operation is not an option if the Hold down option is selected.
Duration for tap down (1-30 s)	The length of time, in seconds, that the Bravo gripper presses down on the filter plate. The duration commences the instant that the gripper begins to press down on the filter plate, not necessarily from the commencement of the vacuum. The Vario vacuum pump has a delay of approximately 3 seconds to start the vacuum after the Tap down begins.
Time allowed to reach pressure *ME4C VARIO Vacuum Pump only*	The length of time, in seconds, during which the pump is allowed to reach the target pressure. The software will display an error message if the target pressure is not reached at the end of the specified time.
Pressure units *ME4C VARIO Vacuum Pump only*	The desired unit of measure: mbar, Torr, hPa, mmHg, cmHg, or inHg. **IMPORTANT** The unit of measure specified in the device profile must match the pressure unit set at the Vario pump in order for the VWorks software to convert to the selected units.

Parameter	Description
Target pressure *ME4C VARIO Vacuum Pump only*	The difference between the pressure of the outside atmosphere above the filter and the pressure in the Vacuum Filtration Station manifold, including the enclosure beneath the filter. For example, if you set the Target pressure to 600 mbar and the ambient pressure displayed on the VARIO pump is 1000 mbar, the vacuum will remain on until the reading on the VARIO pump reaches 400 mbar.
Vent delay (s) *ME4C VARIO Vacuum Pump only*	The length of time, in seconds, to wait for the air pressure under the filter to equalize with the ambient air pressure.
Allow concurrent operation	The option to permit the accessory to operate simultaneously with other tasks.

Example

Goal

On the Bravo Platform, assemble a Vacuum Filtration Station whose configuration is Base–Collection plate–Filter plate–Collar. Filter for 30 seconds. Disassemble the station.

Implementation

In Bravo Diagnostics, configure the profile such that:

- The Vacuum Filtration Station will be assembled at deck location 3.
- The Vacuum Filtration Station base is at deck location 3.
- The collar is at deck location 2.

In the protocol, add the following as shown:

- Filter Plate process
- Collection Plate configured labware (at deck location 5)
- Place Plate (to place the Filter Plate at location 6), Assemble Vacuum, Toggle Vacuum, and Disassemble Vacuum tasks

Figure Example of protocol with Toggle Vacuum (Bravo) task

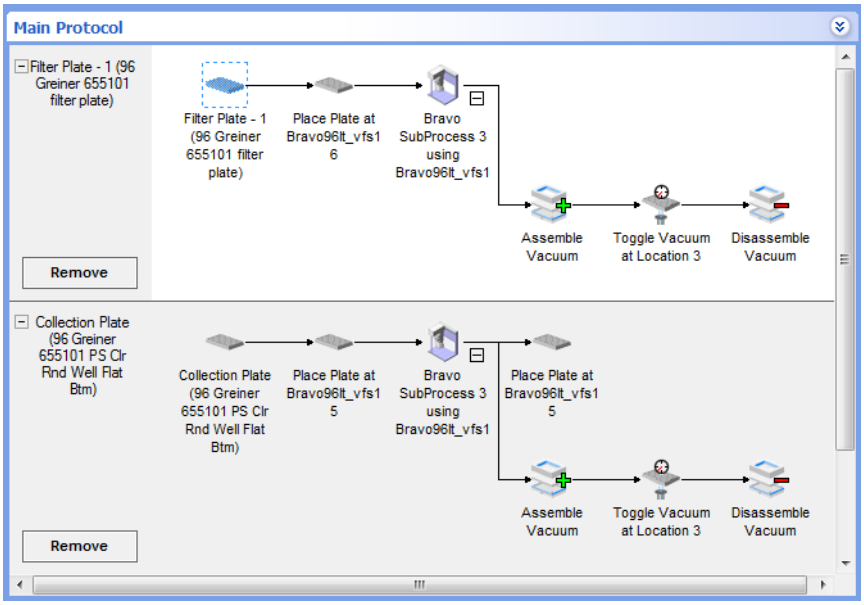


Figure Assemble Vacuum task parameters example

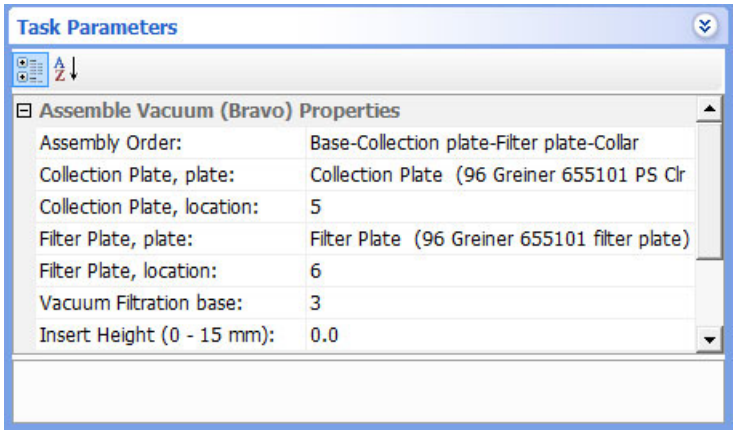


Figure Toggle Vacuum task parameters example

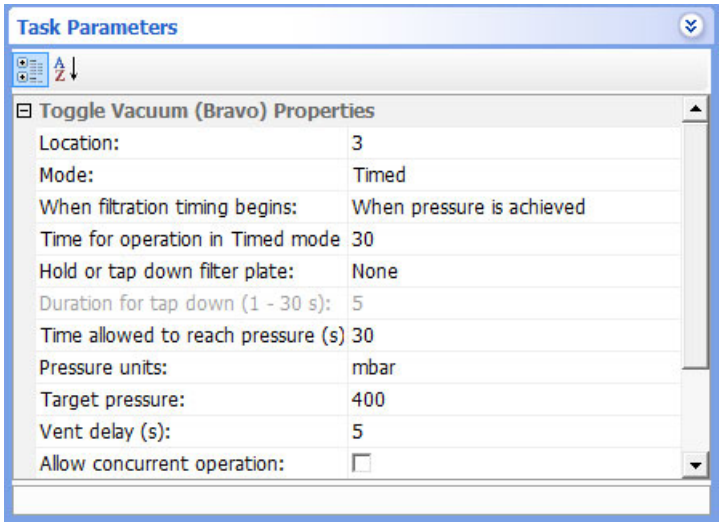
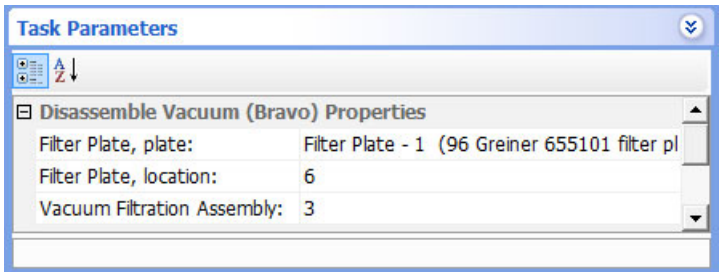


Figure Disassemble Vacuum task parameters example



The resulting protocol will run as follows:

1. Acknowledge that the Filter Plate is starting at deck location 6.
2. Acknowledge that the Collection Plate is starting at deck location 5.
3. Assemble the Vacuum Filtration Station (Base–Collection plate–Filter plate–collar):
   a. Move the Collection Plate from deck location 5 and place it on top of the base at deck location 3.
   b. Move the Filter Plate from deck location 6 and place it on top of the Collection Plate at deck location 3.
   c. Move the collar from deck location 2 and place it on top of the Filter Plate at deck location 3.
4. Turn on the vacuum for 30 seconds, and then turn off the vacuum.
5. Disassemble the Vacuum Filtration Station:
   a. Move the collar back to deck location 2.
   b. Move the Filter Plate back to deck location 6.
   c. Move the Collection Plate back to deck location 5.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Assemble Vacuum task	"Assemble Vacuum (Agilent Bravo)" on page 499
Disassemble Vacuum task	"Disassemble Vacuum (Agilent Bravo)" on page 512
Move and Filter Plate task	"Move and Filter Plate (Agilent Bravo)" on page 568
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

For information about...	See...
Scheduling tasks	"Setting parameters for scheduling tasks" on page 639

Wash Tips (Agilent Bravo)

Description

The Wash Tips (Agilent Bravo) (

) task washes pipette tips using a number of aspirate and dispense actions.

Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Requirements

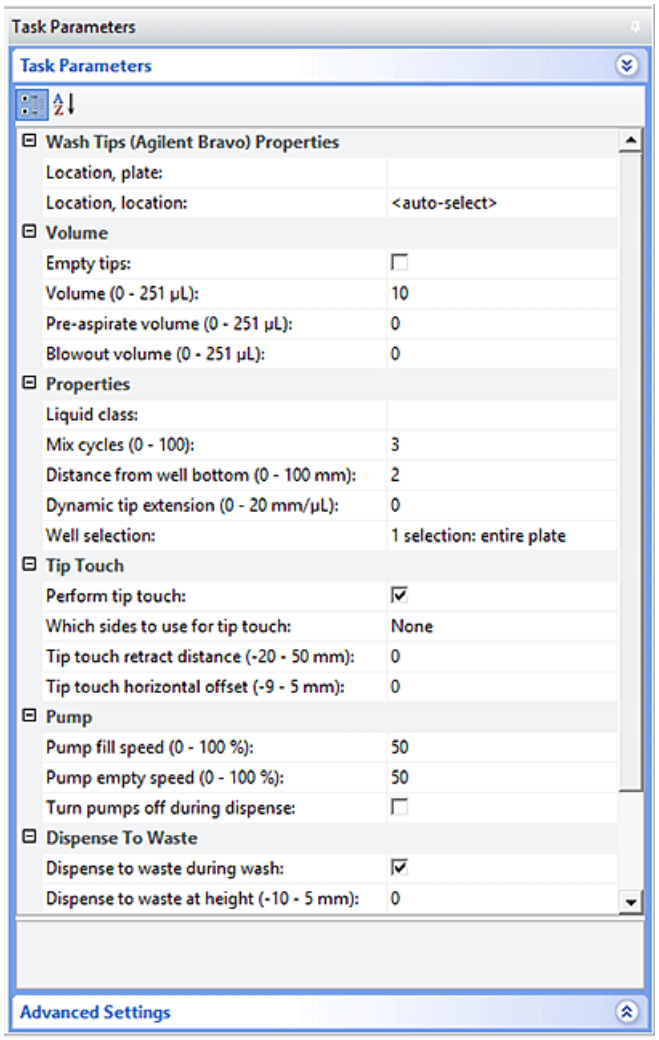
In Bravo Diagnostics ensure that at least one of the following Auto Fill Stations is set up at the correct deck location and associated with a Pump Module.

- Autofilling Reservoir
- MicroWash Reservoir (wash station)
- Open Wash Tray

Ensure that the protocol has a configured labware for the selected autofilling reservoir in addition to the other processes and configured labware required by the protocol.

Task parameters

After adding the Wash Tips task at the desired point in the subprocess, set the following parameters in the **Task Parameters** area:



Parameter	Description
Empty tips	The option to empty the entire contents of the pipette tips, including fluid and air. The Volume parameter is ignored if this option is selected.
Volume (µL)	The volume of liquid to be dispensed from each pipette tip.
Pre-aspirate volume (µL)	The volume of air to be drawn before the pipette tips enter the liquid.
Blowout volume (µL)	Specifies the volume of air to dispense after the main volume has been dispensed. Typically, the blowout volume is the same as the pre-aspirate volume. *Note:* Blowout only occurs in the last quadrant dispensed for a given dispense action.

Parameter	Description
Liquid class	The liquid class associated with this liquid. **IMPORTANT** To ensure consistent pipetting, always select a liquid class for liquid-handling tasks.
Mix cycles	The number of times you want to aspirate and dispense. Each cycle consists of one aspirate action and one dispense action.
Distance from well bottom (mm)	The distance between the end of the pipette tips and the well bottoms during the Wash Tips task. **IMPORTANT** The labware definition must be accurate and the teachpoint must be precise in order for the system to position the tips at the correct distance from the well bottom.
Dynamic tip extension (mm/(µL)	The rate at which the pipette head moves during the Wash Tips task. The software calculates the distance over which the tips will move without crashing. Use dynamic tip extension to prevent spills as the pipette tips displace the liquid. To move the tips: • *At the same rate as the volume change*. Calculate dynamic tip extension (DTE) as follows: DTE = (well depth)/(well vol) = 1/A, where A is the cross-sectional area of a well with straight walls • *Faster than the volume change*. DTE > 1/A • *Slower than the volume change*. DTE < 1/A The starting and ending positions can be calculated as follows: $(V_{dispensed} * DTE) + D_{well\ bottom}$ $(V_{aspirated} * DTE) + D_{well\ bottom}$
Perform tip touch	The option to touch the pipette tip on one or more sides of the well.
Which side to perform tip touch	The wall or walls for tip touch: North, South, East, West, North/South, West/East, West/East/South/North. If you also select the Dispense to waste during wash option, the tip touch is performed on the northeast side only.

Parameter	Description
Tip touch retract distance	The vertical distance the pipette tips rise before touching the sides of the wells.
Tip touch horizontal offset	The horizontal distance the tips move. The value is based on the well diameter specified by the labware definition. The value of the parameter determines the direction of movement: • *0*. Tips move a horizontal distance equal to the well radius. • *Great than 0*. Tips attempt to move past the well radius, which results in a more forceful tip touch. • *Less than 0*. Tips move a distance less than the radius of the well, resulting in a lighter tip touch.
Pump fill speed (%)	The speed, in percent of maximum speed, of liquid flow into the reservoir. For the MicroWash Reservoir, this value should be high enough for the washing liquid to just bubble over the tops of the chimneys.
Pump empty speed (%)	The speed, in percent of maximum speed, of liquid flow out of the reservoir. For the MicroWash Reservoir, this value should be slightly higher than that of the inflow pump to prevent an overflow.
Turn pumps off during dispense	The option to turn off the pumps while dispensing liquid during this task.
Dispense to waste during wash	The option to move the tips by a specified offset (defined in the Labware Editor) and dispense used fluid outside of the reservoir chimney. This option applies only to reservoirs that have chimneys.
Dispense to waste at height (mm)	The height at which the dispense action occurs. For example, during the dispense action, the tips move up to clear the chimneys, move the offset distance, and then lower to the distance you specified. If you want the lower the tips by 10 mm, specify –10 mm.

Example: Wash the pipette tips after Bravo liquid-handling tasks

Goal

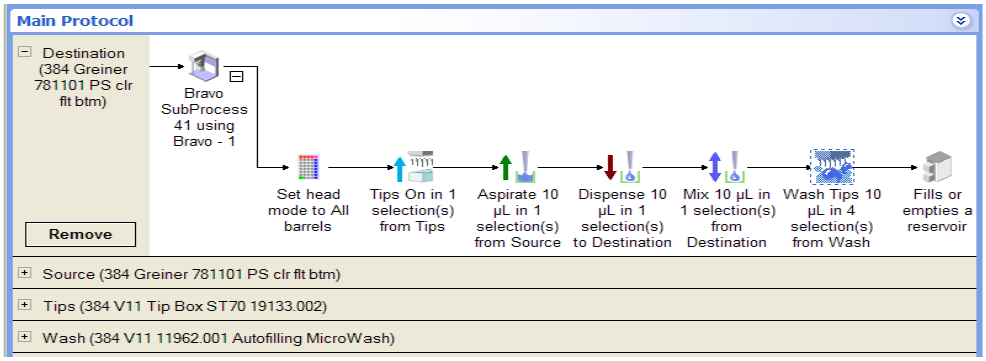
After some liquid-handling tasks, wash the pipette tips in preparation for reuse.

Implementation

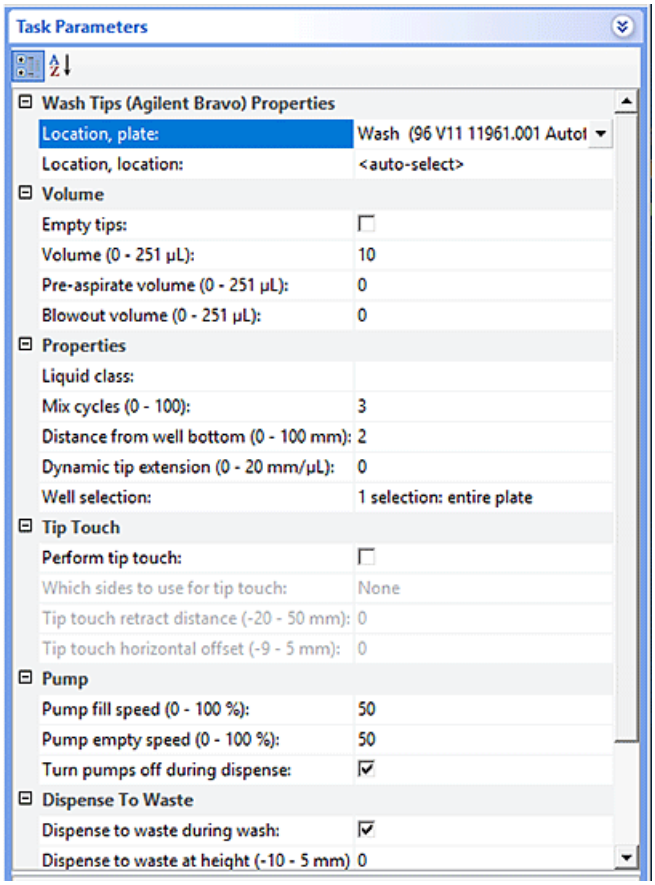
In Bravo Diagnostics, set up an Auto Fill Station accessory (wash station) on a deck location. Associate the reservoir with one or more pump modules.

In the protocol, add a configured labware for the wash station in addition to the other processes and configured labware required by the protocol. In the example shown, the configured reservoir is called Wash.

In the Bravo subprocess where the liquid-handling tasks are specified, a Wash Tips task is added at the end of a cycle of liquid-handling tasks.



In the Wash Tips Task Parameters area, Wash (the name for the wash station) is selected so that the task is performed in this labware.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Aspirate task	"Aspirate (Agilent Bravo)" on page 455
Dispense task	"Dispense (Agilent Bravo)" on page 514
Set Head Mode task	"Set Head Mode (Agilent Bravo)" on page 601
Tips On task	"Tips On (Agilent Bravo)" on page 623
Tips Off task	"Tips Off (Agilent Bravo)" on page 619
Pump Reagent task	"Pump Reagent (Agilent Bravo)" on page 591
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343

For information about...	See...
Microplate-storage tasks	“Setting parameters for microplate storage tasks” on page 411
Scheduling tasks	“Setting parameters for scheduling tasks” on page 639

10 Setting parameters for scheduling tasks

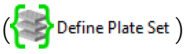
This chapter contains the following topics:

- "Define Plate Set" on page 640
- "Define Variables" on page 645
- "Change Instance" on page 648
- "Group Begin and Group End" on page 650
- "JavaScript" on page 653
- "Loop and Loop End" on page 655
- "Print" on page 658
- "Process Control" on page 666
- "Signal" on page 675
- "Spawn Process" on page 677
- "User Message" on page 682
- "Wait For" on page 686
- "Wait for User (Agilent Bravo)" on page 688

For a description of the Step Started/Completed task, which requires a connection to the Experiments database, see "Configuring a protocol to track experiment data" on page 260.

Define Plate Set

Description

The Define Plate Set task (

) allows you to create an array variable to represent a group of process plates that will be processed using the same tasks in a looping routine. You can reference the plate set using the Location, plate parameter in a task.

Task is available for...	Task is available in...
Any device	Main Protocol, not in a subprocess

Requirements

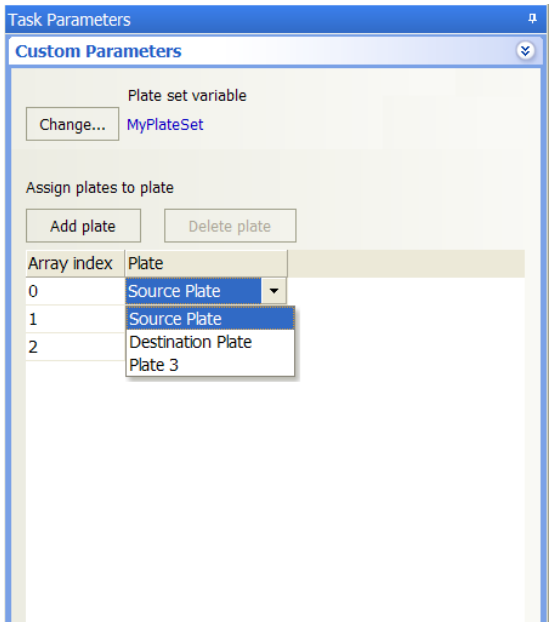
The Define Plate Set task is used with the Loop and Loop End tasks. The Define Plate Set task must be added in the protocol as follows:

- Before the Loop and Loop End tasks that use it.
- In the top-level process, not in the subprocess that contains the Loop and Loop End tasks.
- At the beginning of the process, and the process is not spawned.

The Loop tasks allow you to specify the starting array index value, how frequently the variable will increment or decrement, and the amount by which the index will increment or decrement.

Custom parameters

After adding the Define Plate Set task at the desired point in the protocol, define the plate set in the **Custom Parameters** area:



Parameter	Description
Plate set variable	The name of the variable assigned to the plate set. The name must conform to JavaScript rules for variable names: start with an alphabetical character or the underscore symbol, followed by combinations of alphabetical characters, numbers, and underscore symbols. To define a new name, click **Change,** type the new name, and then click **OK.** **IMPORTANT** Ensure that the Loop task specifies this variable as a Custom Parameter, otherwise the other task parameters will not be able to select from the plate set. For details, see "Loop and Loop End" on page 655.
Add plate	The command that allows you to add a process plate to the plate set. To add a process plate, click **Add plate.** A new row appears in the Array table and an index value is automatically assigned. In the **Plate** field, select a process plate from the list.
Delete plate	The command that deletes the selected process plate.

Parameter	Description
Array table	The list of process plates in the plate set. The table consist of: • *Array index*. The value that identifies the process plate in the array. • *Plate*. The process plate name.

Referencing a plate set in a task

To reference a plate set in a task:

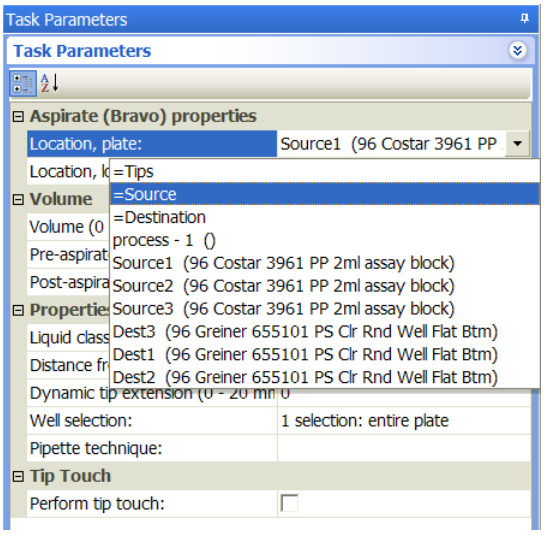
1 Select the task in the protocol.

2 In the **Task Parameters** area, select the plate-set variable from the **Location, plate** list.

If the variable does not appear in the list, make sure:

- The task is inside of a loop.
- The Loop task specifies the plate set name in its Custom Parameters area.

The following example shows the Aspirate task parameters. Three plate-set variables were defined earlier in the protocol and are available for selection: Tips, Source, and Destination. The example shows the Source variable selected.



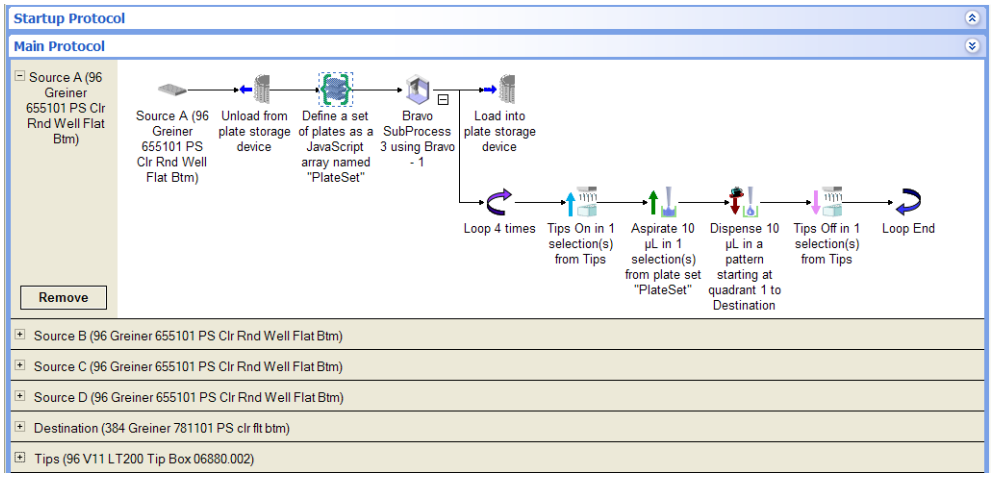
Example: Define a plate set for processing on Bravo Platform

Goal

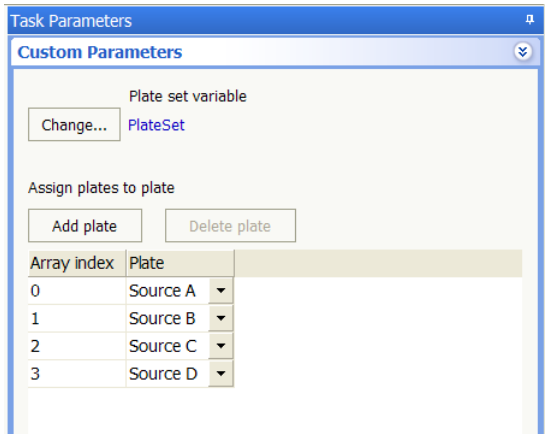
Define a plate set of source microplates (Source A, B, C, and D). Aspirate from each source microplate into a destination microplate.

Implementation

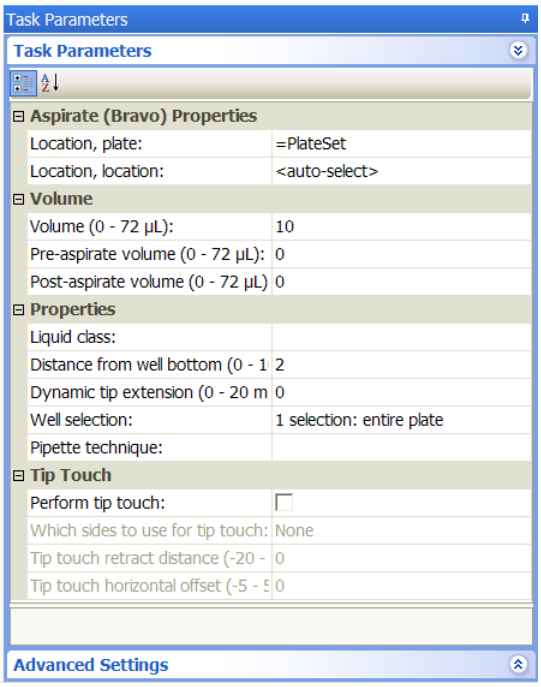
Add a Define Plate Set task at the beginning of the protocol.



The plate set is called PlateSet. This variable is defined in the Custom Parameters for the Define Plate Set task and specified in the Customer Parameters for the Loop task.



Instead of adding four separate Aspirate tasks for each source microplate, add only one Aspirate task in the liquid-handling subprocess. In the Aspirate Task Parameters area, PlateSet is selected.

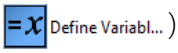


Related information

For information about...	See...
Adding tasks in a protocol	“Adding and deleting tasks” on page 67
Loop and Loop End tasks	“Loop and Loop End” on page 655
Spawn Process task	“Spawn Process” on page 677
Process Control task	“Process Control” on page 666
Task parameter variables	“Using simple variables” on page 93
Microplate-handling tasks	“Setting parameters for microplate-handling tasks” on page 343
Microplate-storage tasks	“Setting parameters for microplate storage tasks” on page 411
Liquid-handling tasks	“Setting parameters for liquid-handling tasks” on page 449

Define Variables

Description

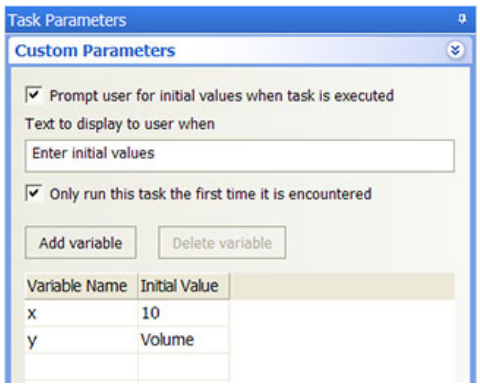
The Define Variables task () allows you to create variables and assign initial values. You can reference the variables in a task.

You can use the Define Variables task with the Loop and Loop End tasks. The Loop tasks allow you to specify the starting variable value, how frequently the value will increment or decrement, and the amount by which the value will increment or decrement.

Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

Custom parameters

After adding the Define Variables task at the desired point in the protocol, define the variables in the **Custom Parameters** area:



Parameter	Description
Prompt operator for initial values when task is run	The option to display all the variables in the Define Variables task and ask the operator to specify initial values for the variables. *Note:* This option is equivalent to the **User data entry into variable** option in the User Message task.
Text to display to user when prompted	The instructions or description that displays in the prompt dialog box.

Parameter	Description
Only run this task the first time it is encountered	The option to run this task the first time it appears in the protocol run. Select this option if the protocol is run multiple times or the task is in a loop, and you only want to run it the first time.
Add variable	The command that adds a new variable in the Variable table. To add a new variable, click **Add variable.** Double-click the **Variable Name** field and type the variable name. Double-click the **Initial Value** field and type the starting value. *Note:* The starting value specified in this task overrides the starting value the operator provides during run time.
Delete variable	The command that deletes the selected variable.
Variable table	The list of variables. The table consists of: • *Variable Name*. The name of the variable. • *Initial Value*. The starting value of the variable.

Referencing variables in a task

To reference variables in a task:

1 Select the task.

2 In the **Task Parameters** area, type the =x in the parameter field, where x is the variable name. For more information about using variables, see "Using simple variables" on page 93.

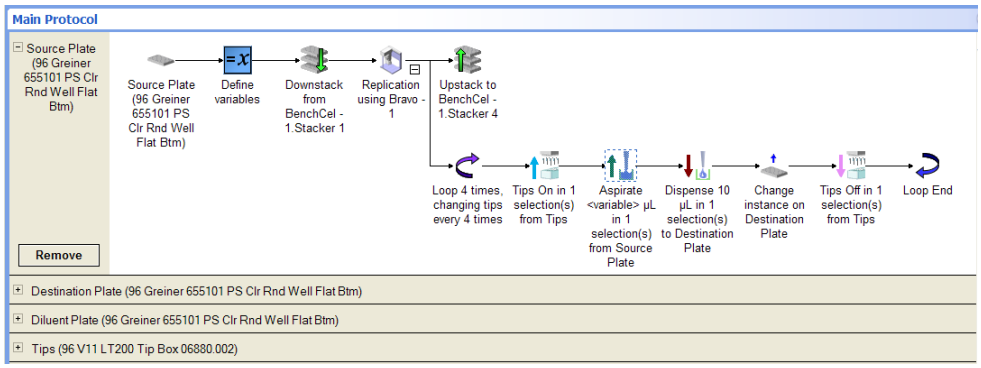
Example: Define a plate set for processing on Bravo Platform

Goal

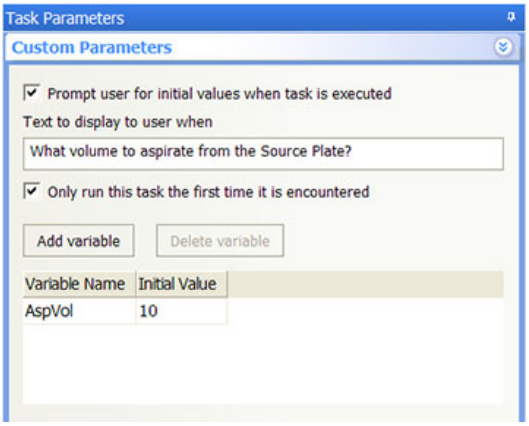
Define a variable to represent the aspirate volume. Request the operator to provide the initial value of the aspirate volume.

Implementation

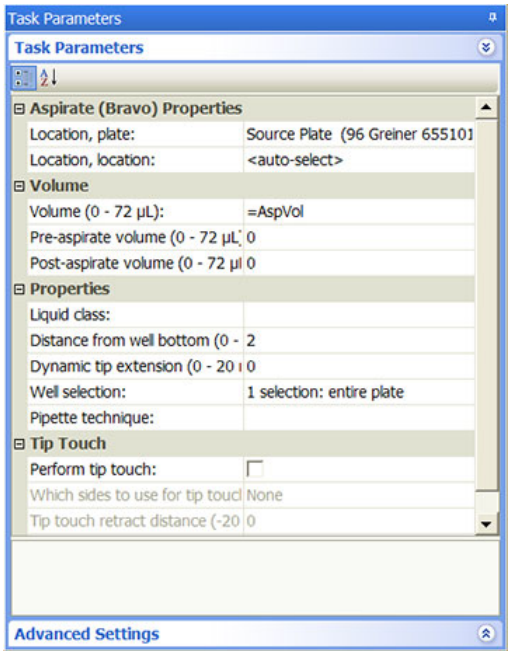
Add a Define Plate Set task at the beginning of the protocol.



The AspVol variable is defined and it represents the aspirate volume. Notice the options are selected to request input from the operator at the start of the protocol run. The operator's input will override the initial value specified in the Custom Parameters area.



In the Aspirate Task Parameters area, the AspVol variable is selected for the aspirate volume.

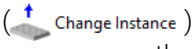


Related information

For information about...	See...
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Loop and Loop End tasks	"Loop and Loop End" on page 655
Spawn Process task	"Spawn Process" on page 677
Process Control task	"Process Control" on page 666
Task parameter variables	"Using simple variables" on page 93
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

Change Instance

Description

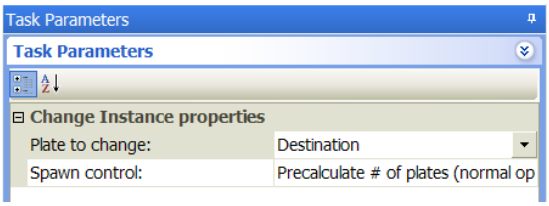
The Change Instance task () allows you to change a plate instance within a loop. For example, you can use the Change Instance task within the Loop and Loop End tasks to aspirate from one source microplate and dispense into different instances of a destination microplate.

Task is available for...	Task is available in...
Any device	Main Protocol

Do not include the Change Instance task in a subprocess of a spawned process that is running as a subroutine. For example, if process 1 spawns process 2 as a subroutine (Spawn as subroutine option) and process 2 contains a subprocess, the Change Instance task will not work in the subprocess.

Task parameters

After adding the Change Instance task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Plate to change	The process plate whose instance you want to change.
Spawn control	The spawn options: • *Precalculate # of plates (normal operation)*. Select this option if the number of microplates can be calculated before the run starts. For example, the calculation can be based on the Number of times to loop value in the Loop task. Do not select this option if the Number of times to loop is a variable, or the looping and the Change Instance task is scripted. • *Spawn new plate when task runs*. Select this option if the Number of times to loop is a variable, or you are scripting the looping and the Change Instance task. The software will not precalculate the number of microplates to use in the Change Instance task. Instead, every time this task runs, a new microplate is brought into the system. • *Do not spawn new plates*. Select this option to script the number of microplates in the run without having to wait for the Change Instance task to execute to bring in new microplates.

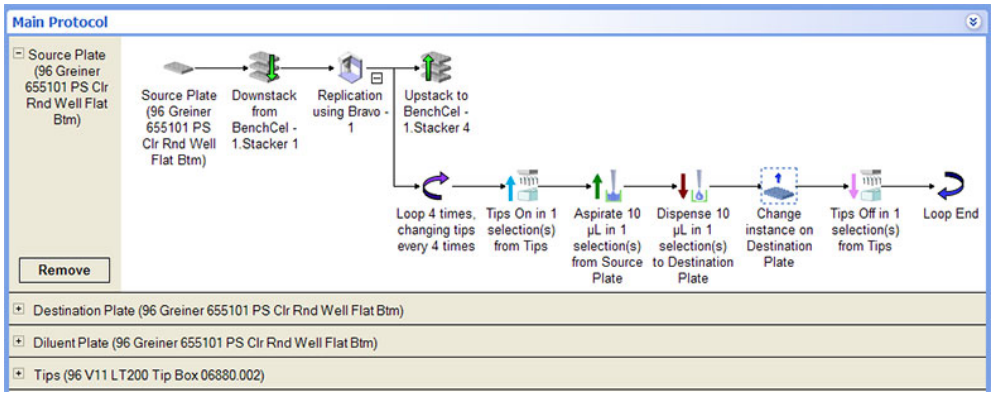
Example: Dispense into multiple destination microplates

Goal

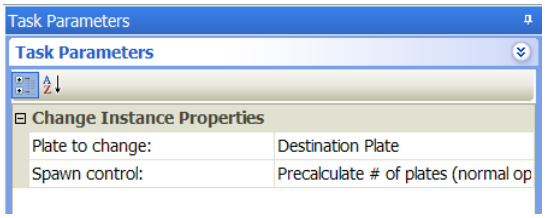
Aspirate from one source microplate and dispense into multiple destination microplates.

Implementation

Within the liquid-handling subprocess loop, add a Change Instance task after the Dispense task.



In the Change Instance Task Parameters area, the Destination Plate is selected.



Related information

For information about...	See...
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Loop and Loop End tasks	"Loop and Loop End" on page 655
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

Group Begin and Group End

Description

The Group Begin (Group Begin) and Group End (Group End) tasks are used together to:

- Group a number of tasks within a subprocess.
- Constrain the grouped tasks to run in the specified sequence.
- Prevent downstream tasks from starting until the grouped tasks are finished.

Note: If the tasks are not grouped, the software will determine the fastest task sequence based on available resources.

During the protocol run, the software will make sure the labware needed by the grouped tasks are on the device locations before running the first task in the group. In addition, while the grouped tasks are running, only the microplates required by the grouped tasks will be allowed to be moved onto, off of, or within the device.

Use the Group Begin and Group End tasks if you want to control the sequence of liquid-handling tasks to make sure each microplate is handled the same way in the same sequence.

Note: To further control the tasks, especially those with time limits, you can specify time constraints between dependent tasks. See "Specifying time constraints between dependent tasks" on page 71.

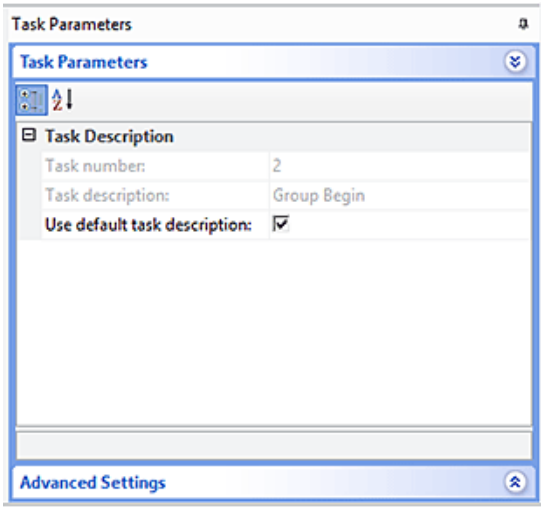
Task is available for...	Task is available in...
Bravo Platform	Main Protocol, Bravo Subprocess

Requirements

You cannot add a Change Instance task within the grouped tasks.

Task parameters

After adding the Group Begin and Group End tasks at the desired point in the protocol, the following Task Parameters are set by default. You can edit the Task description as desired.



Parameter	Description
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.

Parameter	Description
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

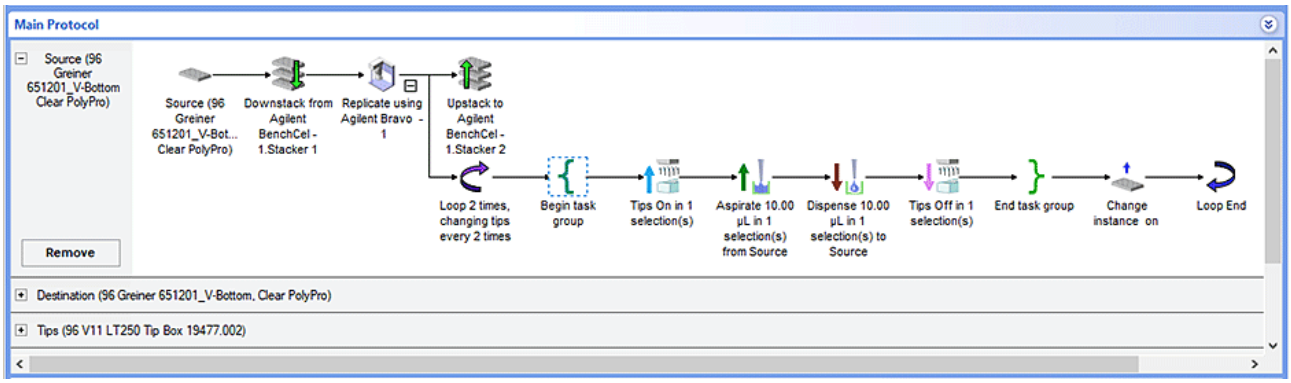
Example: Group liquid-handling tasks

Goal

Make sure all liquid-handling tasks are finished before incubation begins.

Implementation

Add the Group Begin and Group End tasks around the liquid-handling tasks. Notice that the Change Instance task is not within the group.



Related information

For information about...	See...
Adding tasks in a protocol	“Adding and deleting tasks” on page 67
Microplate-handling tasks	“Setting parameters for microplate-handling tasks” on page 343
Microplate-storage tasks	“Setting parameters for microplate storage tasks” on page 411
Liquid-handling tasks	“Setting parameters for liquid-handling tasks” on page 449

JavaScript

Description

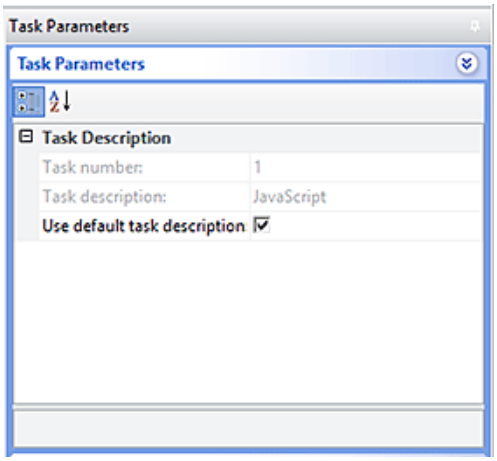
The JavaScript task (JavaScript) runs the specified JavaScript at the desired point in the protocol. Typically, you use the JavaScript task to run a program that is independent of any task.

Note: If you want to run a JavaScript that skips a task or changes the parameters of the task, write the JavaScript code in the Advanced Settings area of that task. See "Using JavaScript" on page 100 for more information.

Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

Task Parameters

The following Task Parameters are set by default. You can edit the Task description as desired.

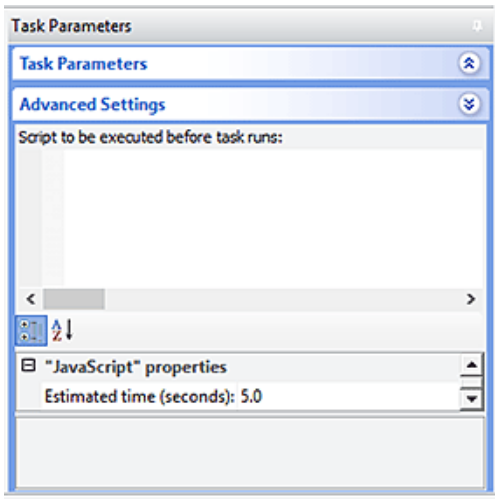


Parameter	Description
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

Advanced Settings

Write the JavaScript code in the **Advanced Settings** area.

Note: The Estimated time (seconds) property is used for simulation. For details, see "Simulating the protocol run" on page 80.



Related information

For information about...	See...
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Writing JavaScript in a protocol	"Using JavaScript" on page 100
Define Variables task	"Define Variables" on page 645
Task parameter variables	"Using simple variables" on page 93
Adding start and finish protocol scripts	"Using start and finish protocol scripts" on page 152
JavaScript utilities	"Using JavaScript utilities" on page 153
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

Loop and Loop End

Description

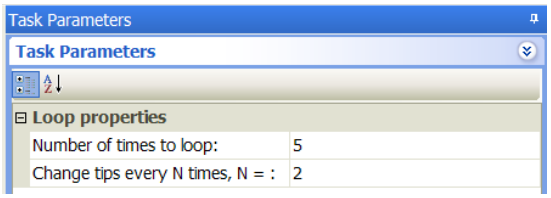
The Loop (Loop) and Loop End (Loop End) tasks are used together to repeat a set of tasks that are within the two tasks.

Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

Task parameters

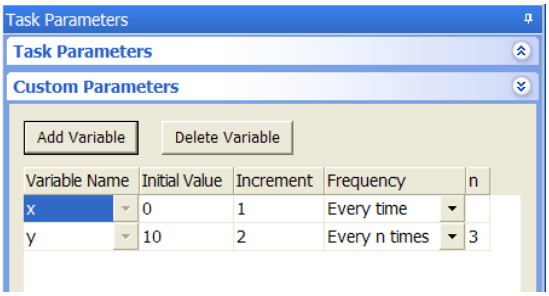
Note: Loop End does not have any task parameters.

After adding the Loop task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Number of times to loop	The number of times you want to run the tasks within the loop.
Change tips every N times, N=	The number of times to run the tasks in the loop before changing the tips. For example, if N is set to 2, the Tips On task will be run every odd number of the loop: first, third, fifth, and so on. The Tips Off and Change Instance tasks that operate on the tip box used by the Tips On task will run every even number of the loop: second, fourth, sixth, and so on. *Note:* If the loop does not contain any tasks that require pipette tips, the software ignores this parameter.

If you want to change the value of variables during the looping process, click **Custom Parameters** and set the following:



Parameter	Description
Add Variable	The command that adds a variable in the Variable table. To add a variable, click **Add Variable.** A blank row appears in the table. Select the desired value for each column.
Delete Variable	The command that deletes the selected variable from the table.
Variable table	The table that lists the variables to be incremented or decremented in the loop.
Variable name	The variable you want to increment or decrement. Select the variable from the list. The variables are defined in the Define Variables task, the Define Plate Set task, or the JavaScript code in the Advanced Settings area. If the variable is new and was not previously defined elsewhere, and an initial value is given, the variable will be defined the moment the Loop task runs.
Initial value	The starting value to be used the first time through the loop. For variables, the Initial value is the starting value of the variable. For plate sets, the Initial value is the starting array index value. *Note:* The starting variable value specified in this task overrides the starting variable value the operator provides during run time.
Increment	The amount you want the variable to increment or decrement.

Parameter	Description
Frequency	The frequency of the increment or decrement: • *Every time*. The variable should increment or decrement every loop. • *Every n times*. The variable should increment or decrement every specified number of times. • *After first time*. The variable should increment or decrement starting from the second loop. • *After last time*. The variable should increment or decrement after the last loop.
n	The number of times to loop before a variable increments or decrements. This field is available only if you selected Every n times.

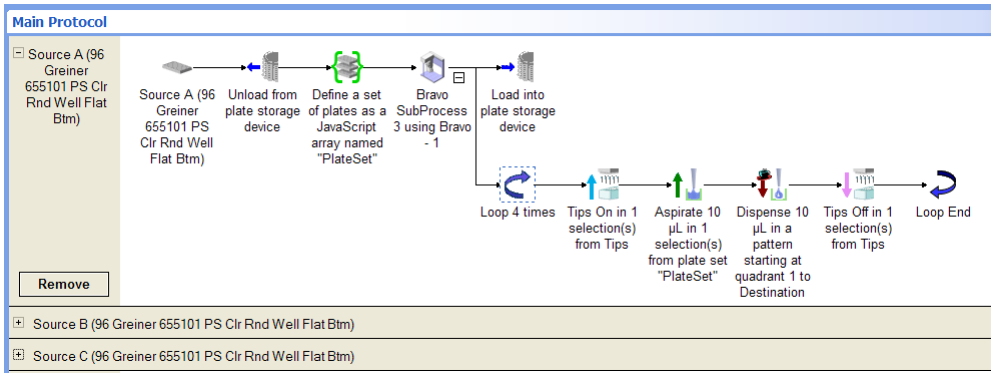
Example: Use the Loop task for plate set indexing

Goal

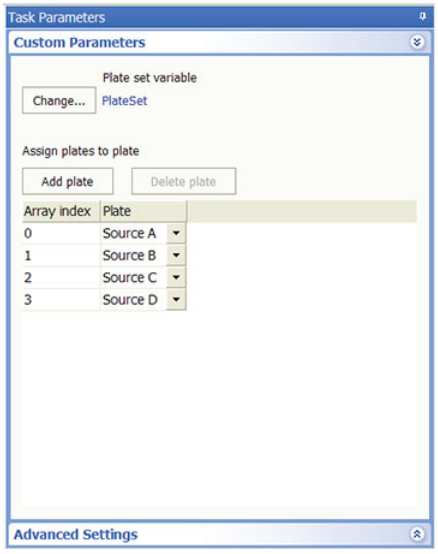
In a protocol where a plate set variable is defined, use the Loop task to specify the starting plate set (or array) index, how frequently the variable will increment or decrement, and the amount by which the index will increment or decrement.

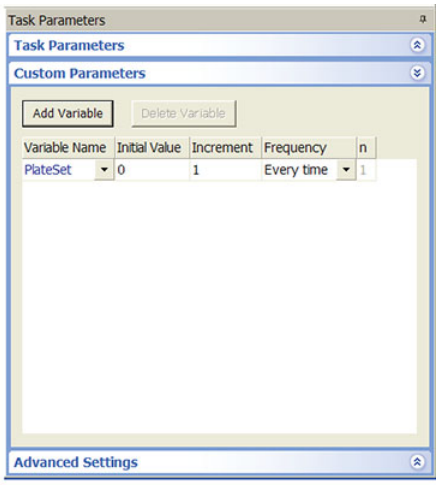
Implementation

In the following protocol example, a plate set is defined and used in the liquid-handling tasks.



In the Define Plate Set Task Parameters area, an array of source microplates is defined and indexed (0, 1, 2, and 3). In the Loop Custom Parameters area, the PlateSet variable is selected. The Initial Value is set to 0 to match the first PlateSet index (0). The PlateSet variable will increment by 1 every time through the loop. So the first time through the loop, PlateSet index 0, or SourceA is processed. The second time through the loop, PlateSet index 1, or Source B is processed, and so on.





Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Define Plate Set task	"Define Plate Set" on page 640
Task parameter variables	"Using simple variables" on page 93
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

Print

Description

The Print task (Print) prints barcode labels using the Microplate Labeler. The task does not apply labels to labware.

Use this task if you want to print labels and manually apply the labels to labware such as tubes or other containers. You can also apply the labels to pages in your lab notebook for record keeping purposes.

Task is available for...	Task is available in...
Microplate Labeler (standalone only)	Startup Protocol Cleanup Protocol

Requirements

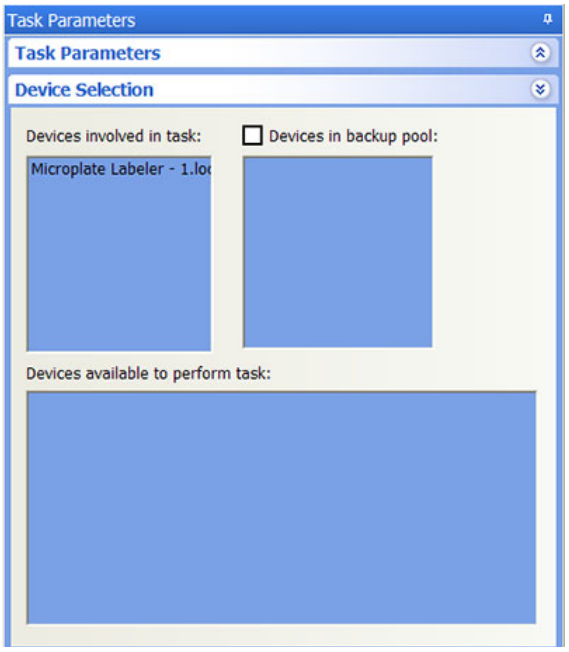
The requirements for the Print task are identical to the requirements for the Print and Apply task. See "Print and Apply" on page 381.

Selecting devices for the task

You must select a device for the Print task before you can set the task parameters. After adding the Print task in the protocol, select the task, and then click **Device Selection** in the **Task Parameters** area.

To select a device for the task:

Double-click the desired device in the **Devices available to perform task** area to move it to the **Devices involved in task** area.



Setting the task parameters

IMPORTANT

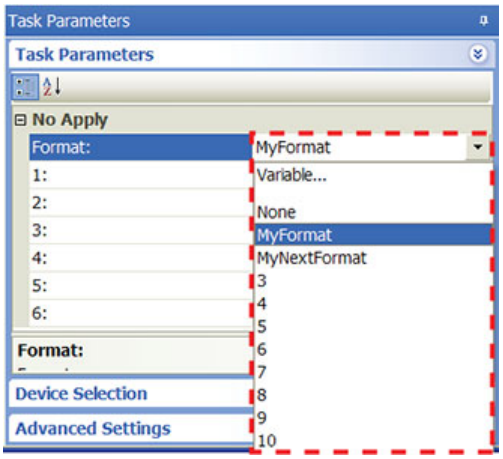
Make sure the label formats are uploaded to the printer. Be sure to initialize the Microplate Labeler device before you set the task parameters.

After selecting the device to use for the Print and Apply task, you can set the parameters in the **Task Parameters** area. The area lists the four sides of a microplate (south, west, north, and east). For each side, you can select a label format and specify the data that will substitute for the text and barcode fields in the label format.

CAUTION **Format selection and field information are saved with the protocol. If the formats on the printer are changed, initializing the device will overwrite the information in the protocol. For example, suppose you created a protocol and selected a format called MyFormat, and later, MyFormat was deleted from the printer. The next time you initialize the device and open the protocol, MyFormat will not appear in the protocol.**

To set the task parameters:

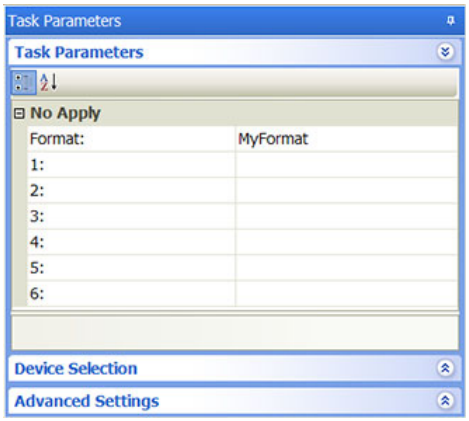
1 Select the barcode format in the **Format** list:



Format selection	Description
None	Indicates no label will be printed.
Format name or number	Uses a format that was set up in Microplate Labeler Diagnostics. *Note:* If you do not see a list of formats, make sure the label formats are not empty (formats must contain at least one field), the formats are uploaded to the printer, and the Microplate Labeler device is initialized.

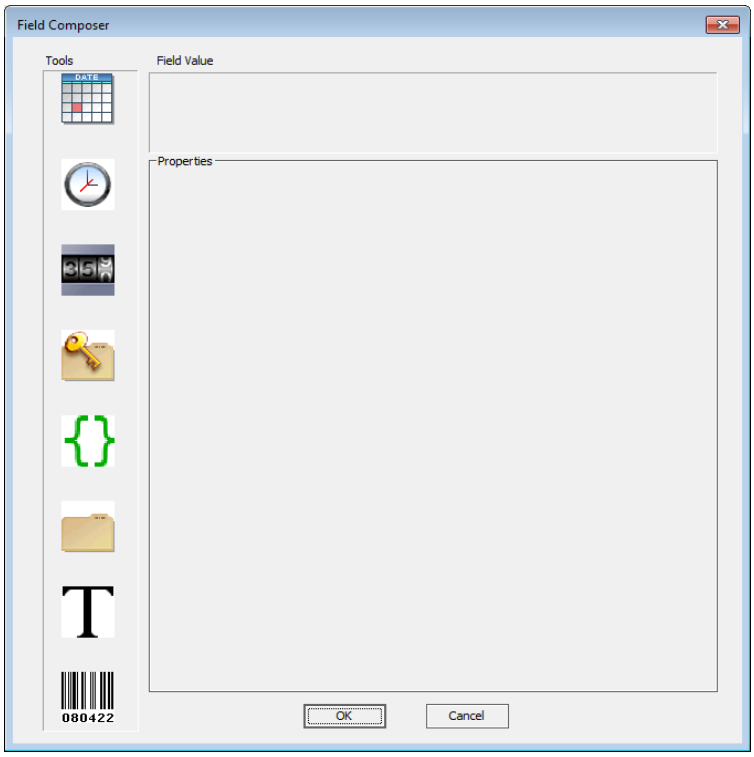
As soon as you select a format, fields appear in the Task Parameters area. You can specify the information you want to print in these fields.

Note: The number of fields that appear depends on the format you select.



2 Click a field, and then click the ··· button that appears. The Field Composer dialog box opens.

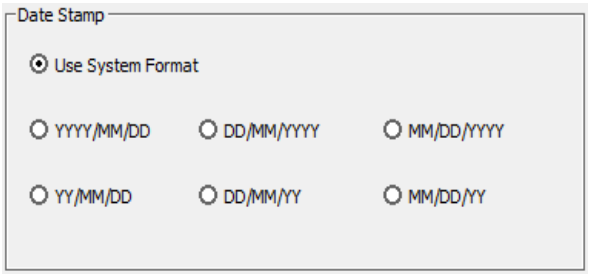
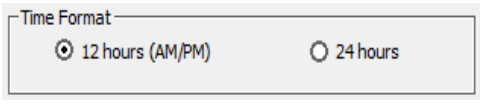
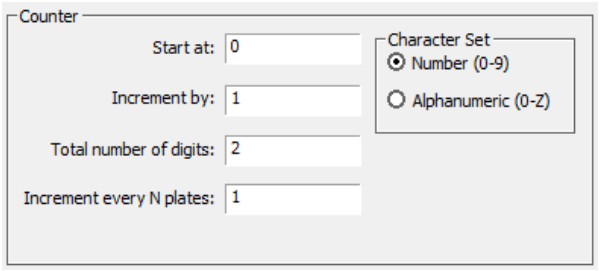
The Field Composer allows you to specify the information to print on the barcode label. For example, you can print the current date and time.

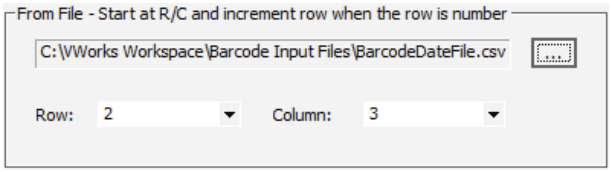
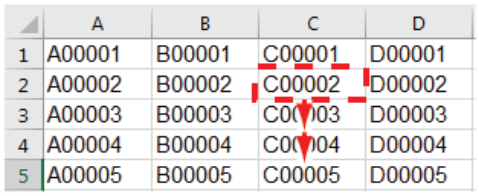
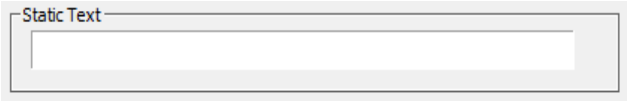


3 In the **Tools** area, double-click one or more of the following icons to specify the information to be printed on the barcode label. The selected icon appears in the **Field Value** area.

IMPORTANT

For field limitations, such as the maximum number of characters permitted or symbology-dependent limitations, check the format you set up in Microplate Labeler Diagnostics. See also the *Microplate Barcode Labeler User Guide*.

Icon	Description
	Prints the current date. Click the icon in the **Field Value** area. In the **Properties** area, select the desired date format. YYYY is the year, MM is the month, and DD is the day. *Note:* The Use System Format option uses the local computer's date format. 
	Prints the current time. Click the icon in the **Field Value** area. In the **Properties** area, select the desired data format: **12 hours (AM/PM)** or **24 hours.** 
	Prints a numeric or alphanumeric value that can be incremented. Click the icon in the **Field Value** area. Set the following in the **Properties** area: • *Character Set*. The option to use either numeric or alphanumeric characters. • *Start at*. The starting value. • *Increment by*. The amount by which the value increments. • *Total number of digits*. The total number of digits or characters, including leading 0s. • *Increment every N plates*. The increment value. For example, 1 increments the value every microplate. 

Icon	Description
	Not used. If you add this icon, the software will ignore it.
	Prints the values in the specified barcode input file. Click the icon in the **Field Value** area. In the **Properties** area, click the ··· button to select the input file. Specify the row and column number of the starting value. During the run, the software automatically increments to the next row to print the next value. For example, Start at row is set at 2, Start at col is set at 3. During the run, the software starts with the value in row 2 column 3 (C00002). Then, the software moves to row 3 column 3 (C00003), row 4 column 3 (C00004), row 5 column 3 (C00005), and so on.  
	Prints the text you specify. Click the icon in the **Field Value** area. In the **Properties** area, type the text you want to print on every microplate. 
	Not used. If you add this icon, the software will ignore it.

4 When you are finished, click **OK.** The information you specified appears in the Task Parameters area.

Example

Goal

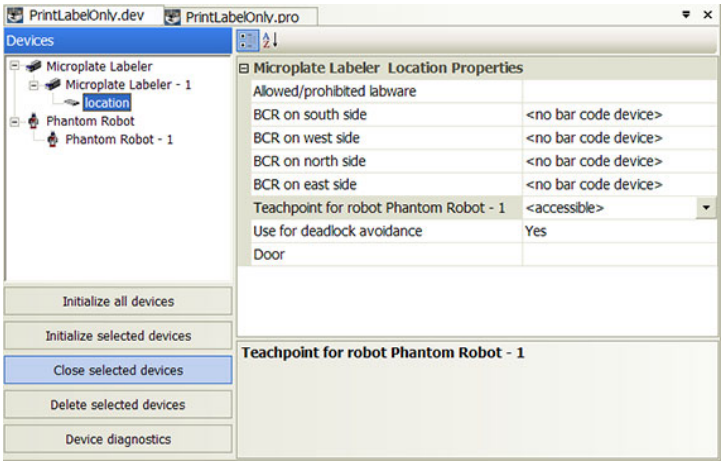
Using the Microplate Labeler, print values read from a barcode data file. The software should start reading the file from row 1 column 1.

The software should pause after it prints a label to permit you to remove the label from the device and manually apply it to a page in your lab notebook.

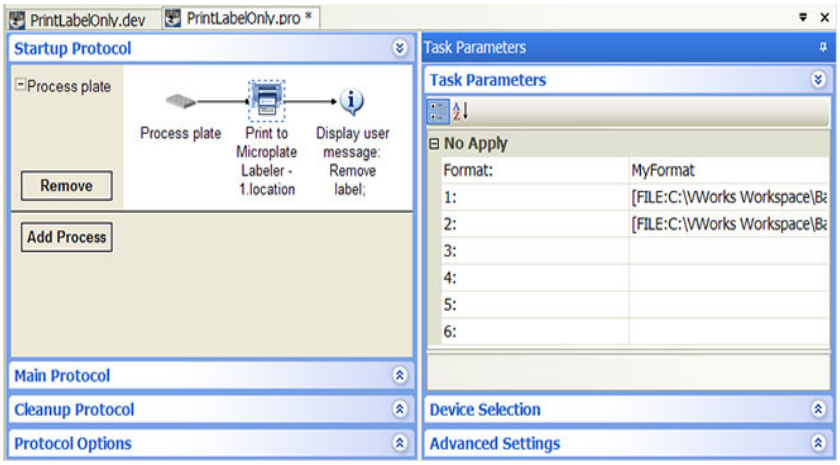
Implementation

Note: This example assumes that the Microplate Labeler is set up correctly and the format, MyFormat, is already defined and loaded to the printer. MyFormat contains two fields. Field 1 is a human-readable text field. Field 2 is a barcode field.

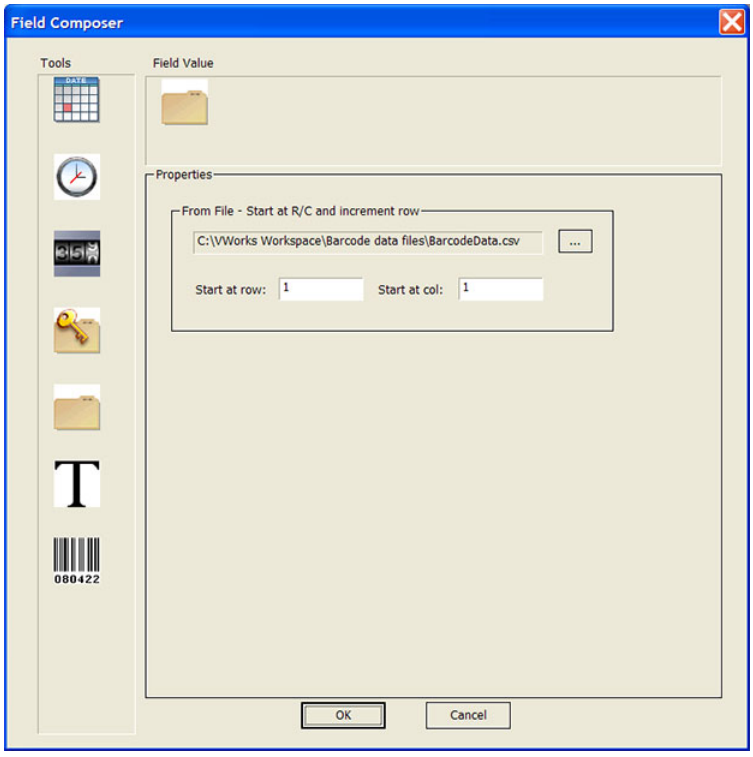
Create a device file that contains both the Microplate Labeler and a Phantom Robot, as shown. In the **Microplate Labeler Location Properties** area, make sure the **Teachpoint for Phantom Robot** is **<accessible>.**



Create a Startup Protocol as shown and add a Print task and a User Message task. The User Message task creates the pause to permit you to remove the label from the device. When setting the Print and Apply task parameter, select MyFormat.



For each of the two format fields (1 and 2), open the Field Composer dialog box and double-click the data file icon in the **Tools** area to add it to the Field Value area. In the **Properties** area, locate and select the data file. In the **Start at row** box, type 1. In the **Start at col** box, type 1.

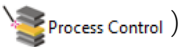


Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Print and Apply task	"Print and Apply" on page 381
Specifying the barcode input file directory	"Setting protocol options" on page 42
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

Process Control

Description

The Process Control task () is used to initiate a specific process within the same protocol based on an upstream condition or the type of labware entering the process.

Task is available for...	Task is available in...
Any device	Main Protocol

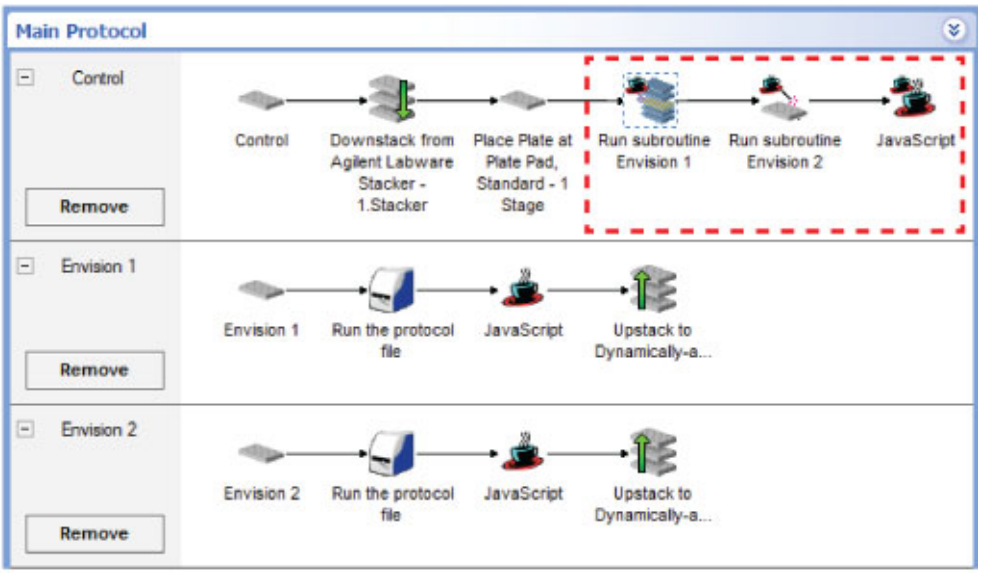
When to use Process Control instead of Spawn Process

The Process Control task extends the functions of a Spawn Process task with simplified implementation in a protocol. Instead of adding multiple Spawn Process tasks in one process, you can simplify the protocol by adding a single Process Control task.

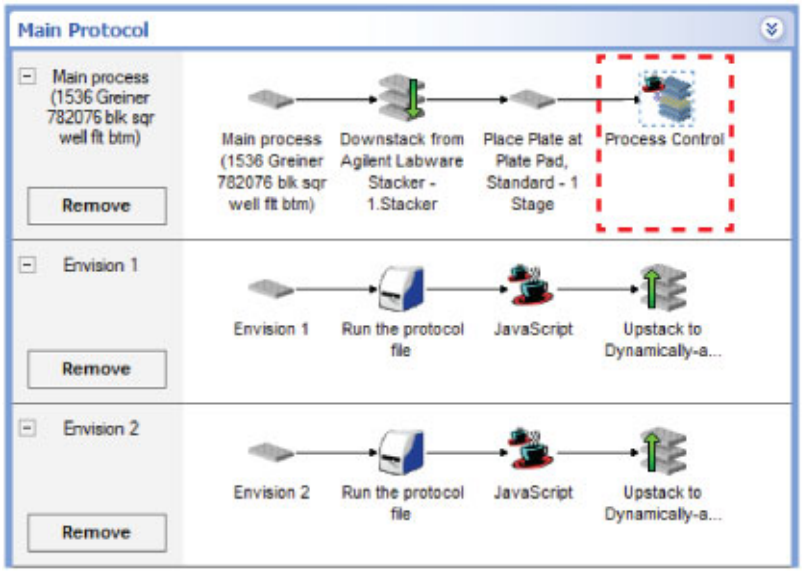
For example, suppose you have two devices in a device pool. You want to design the protocol such that if the first device is busy, the system will use the second device.

You can write the protocol in one of two ways:

- Add two Spawn Process tasks, one for using each device. The required JavaScript will be distributed across the two Spawn Process tasks and a separate JavaScript task, thus making debugging and future updates difficult.

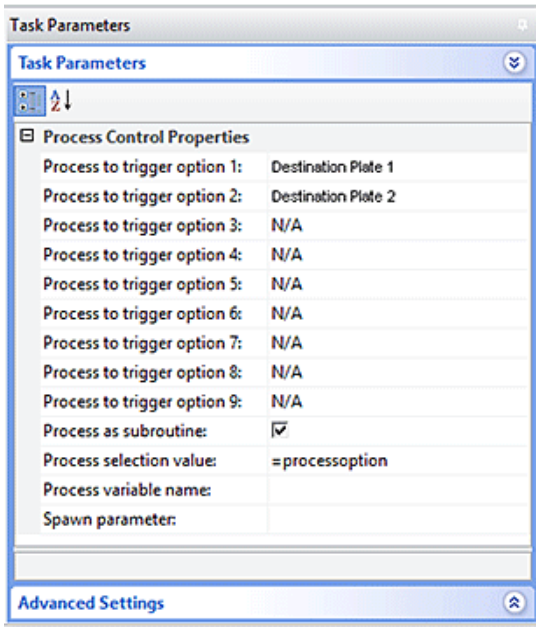


- Add a single Process Control task that controls the use of both devices. The required JavaScript is consolidated in the Process Control task, thus facilitating debugging and future updates.

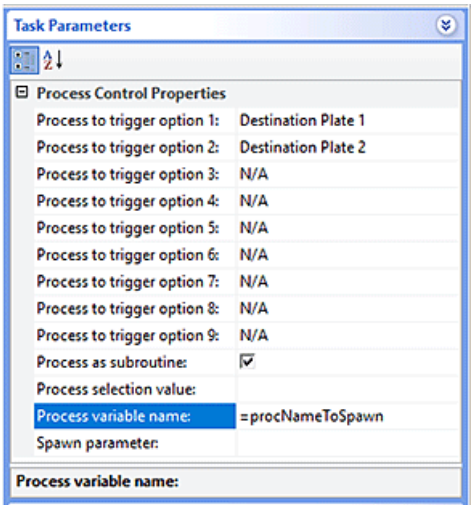


Task parameters

After adding the Process Control task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Process to trigger option 1—9	One of the processes that will be initiated when the run reaches the Process Control task. Select the process that you want to initiate. For example, if you have two potential processes to initiate, Destination1 and Destination2, select Destination1 for Process to trigger option 1 and Destination2 for Process to trigger option 2. One of the following options determine which process will be initiated: • If the **Process variable name** is blank, the **Process selection value** determines which process will be initiated. • If the **Process variable name** specifies a name, the **Process selection value** is not used, and instead the first **Process to trigger option** with a process name that matches the **Process variable name** determines which process to spawn.
Process as subroutine	The option to run the initiated process as a subroutine (or subprocess) of the current process. Select the check box if you have data, such as barcode information, from the current process that you want to pass to the initiated process. Clear the check box if the two processes can run in parallel when the current process reaches the Process Control task. In this case, the task will use the **Spawn parameter** property if it is specified.
Process selection value	The value that determines which process to initiate if the **Process variable name** is blank. For example, if you specify 5, then the process you selected for Process to trigger option 5 will initiate. You can also specify a variable. During the protocol, the value of the variable (1, 2, 3,..., or 9) is passed to the Process Control task from an upstream task or from the Process Control task JavaScript. The value determines which process to run. If the value is 1, the specified process for **Process to trigger option 1** will initiate; if the value is 2, the specified process for **Process to trigger option 2** will initiate; and so on.

Parameter	Description
Process variable name	The name of the process to initiate. Do one of the following: • Leave this property blank if you want the **Process selection value** to determine which process to initiate. For example, in the preceding Task Parameters figure: Process selection value =processoption In this example, the process to be launched will be determined by the processoption JavaScript variable's numeric value. If processoption is 1, the Destination Plate 1 process will be spawned. If processoption is 2, the Destination Plate 2 process will be spawned. • Specify a name that matches the name of the process in the **Process to trigger option** that will determine which process to spawn. Use this option if you want to specify the process to spawn by its name, rather than by its number. For example, suppose you had a JavaScript variable named procNameToSpawn, which through scripting contains a string that contains another process name, such as Destination Plate 1. In this case, you could set the following: Process variable name =procNameToSpawn  In this case, the **Process selection value** will be ignored, and the process spawned will be: – Destination Plate 1 (Process to trigger option 1) if procNameToSpawn is Destination Plate 1, or – Destination Plate 2 (Process to trigger option 2) if procNameToSpawn is Destination Plate 2 *Note:* If procNameToSpawn is set to any string other than the name of a process that appears in the **Process to trigger option** list, the task will instead use the **Process selection value** to determine which process to spawn.

Parameter	Description
Spawn parameter	The option to enter a string, used by a script, to identify which process spawned a given spawned process. In the spawned process or processes, `plate.spawnParameter` will be set to whatever value appears in the **Spawn parameter** property. This property can be set to a scripted value or set in the task's script. *Note:* If **Spawn a subroutine** is selected, **Spawn parameter** is not used.

Example

Goal

Downstack microplates from the Labware Stacker, scan the microplate barcode, read the microplate using an available Envision Reader, and then upstack the microplate to an available Labware Stacker.

Use the pool of two Envision Readers during the protocol run:

- If the first reader is busy, use the second reader.
- If the second reader is busy, use the first reader.
- If both readers are busy, wait 2 seconds and check each reader again.

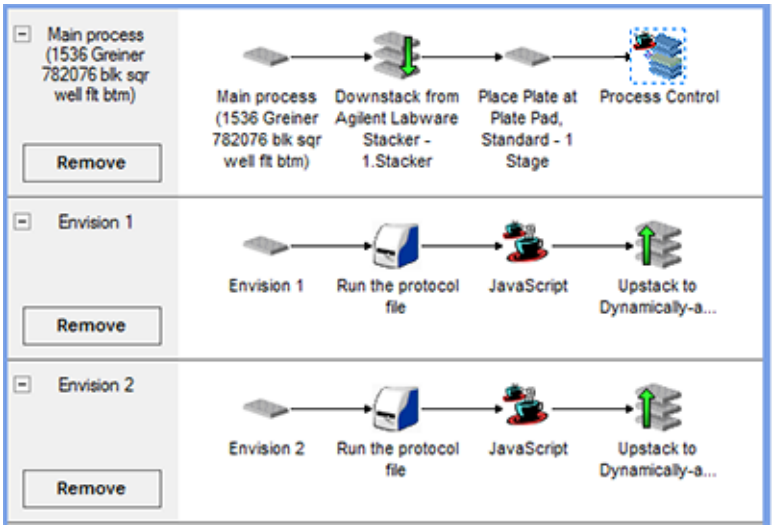
Set variables for the following purposes:

- Determine which Envision Reader to use. In the example, the variable used for this purpose is `procoption`.
- Determine the state of the first Envision Reader. Valid values are true (busy) and false (available for use). In the example, the variable used for this purpose is `bEnvision1Busy`.
- Determine the state of the second Envision Reader. Valid values are true (busy) and false (available for use). In the example, the variable used for this purpose is `bEnvision2Busy`.

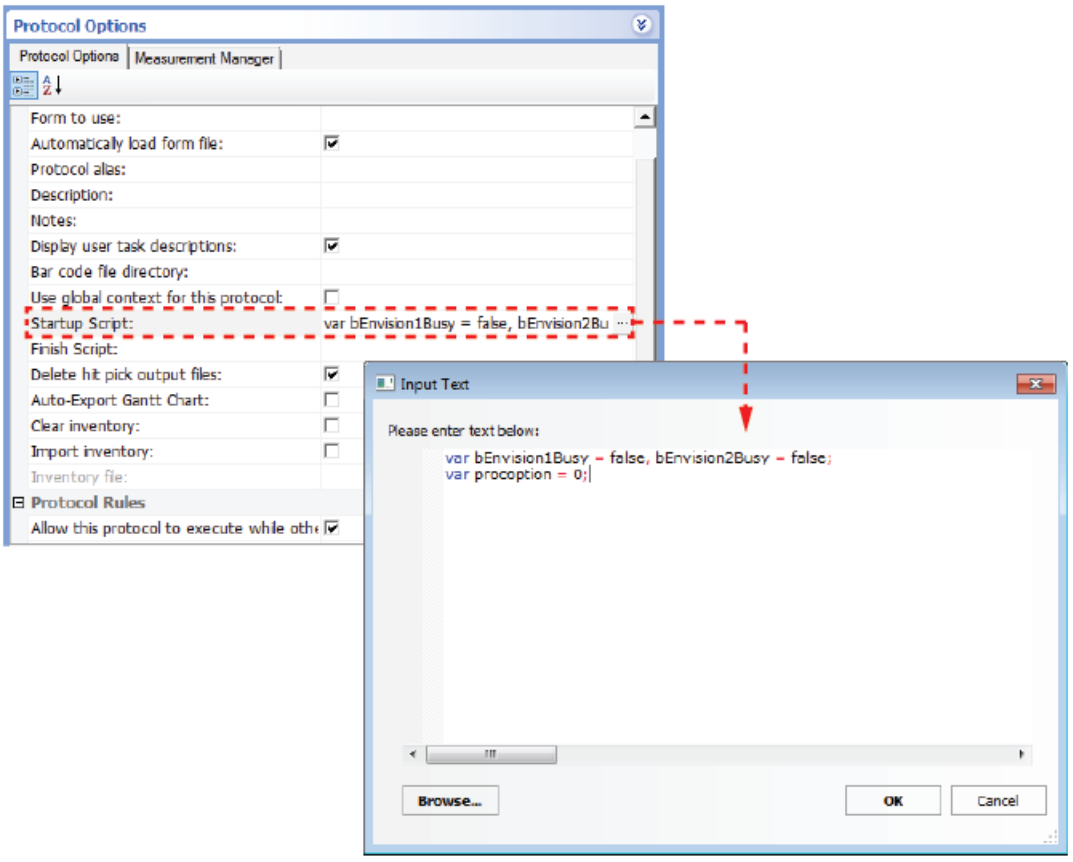
Implementation

Create a protocol as shown.

Note: The barcode reader is attached to the platepad, so a Place Plate task is added to scan the microplate barcode at the platepad.

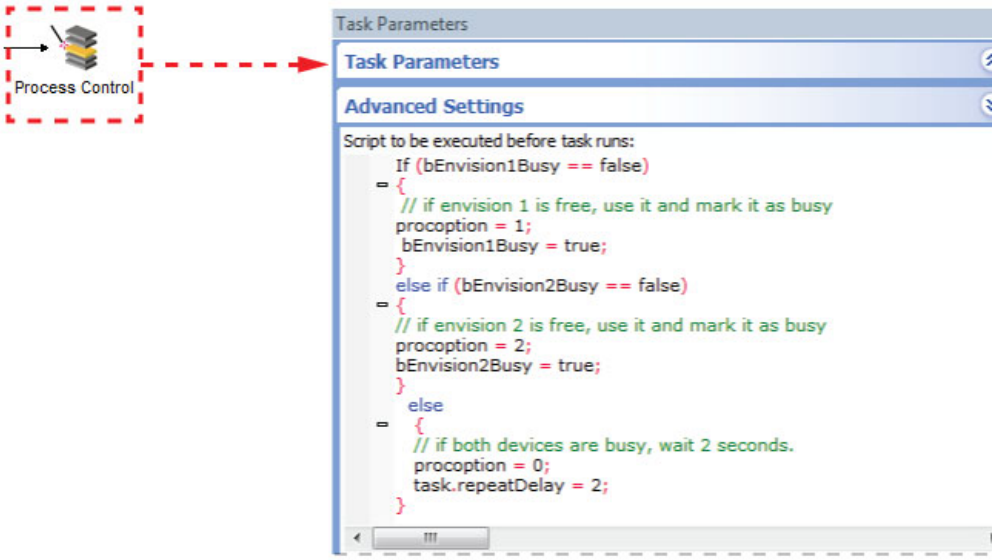


In the **Protocol Options Startup Script**, add JavaScript to reset all relevant variables at the start of the run. Set the Envision Reader busy state (`bEnvision1Busy` and `bEnvision2Busy`) to false, and set the `procoption` to 0.



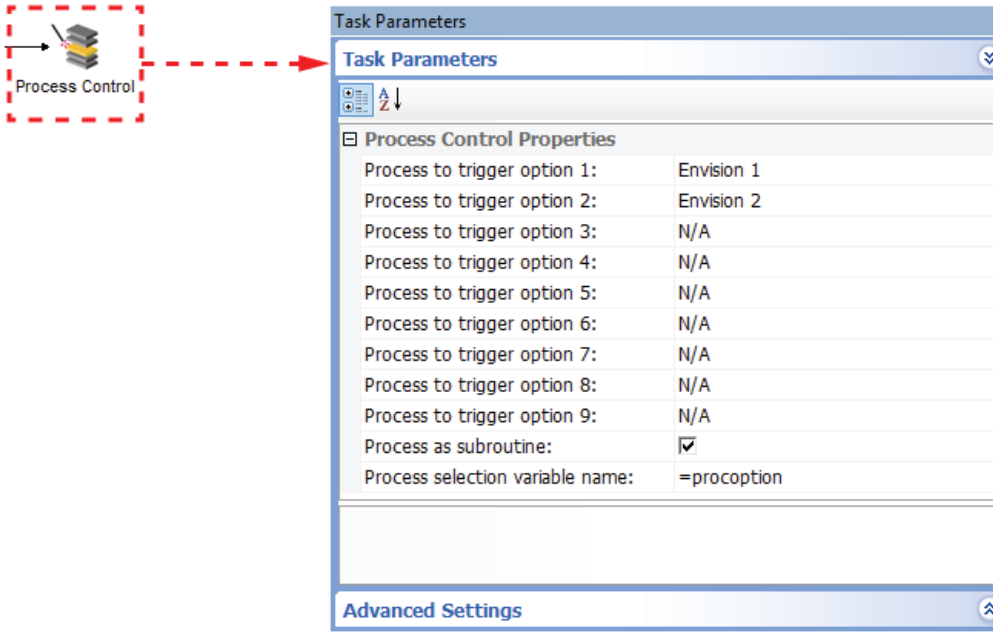
Before setting the task parameters, add JavaScript to the **Process Control** task to determine which Envision Reader to use. (The script is run before the Process Control task is performed.) The script logic is as follows:

- The system checks to see if the first Envision Reader is available. If it is, `procoption` is set to 1.
- If the first Envision Reader is busy, the system checks the second Envision Reader. If it is available, `procoption` is set to 2.
- If both devices are busy, the system will wait 2 seconds before checking the devices again.

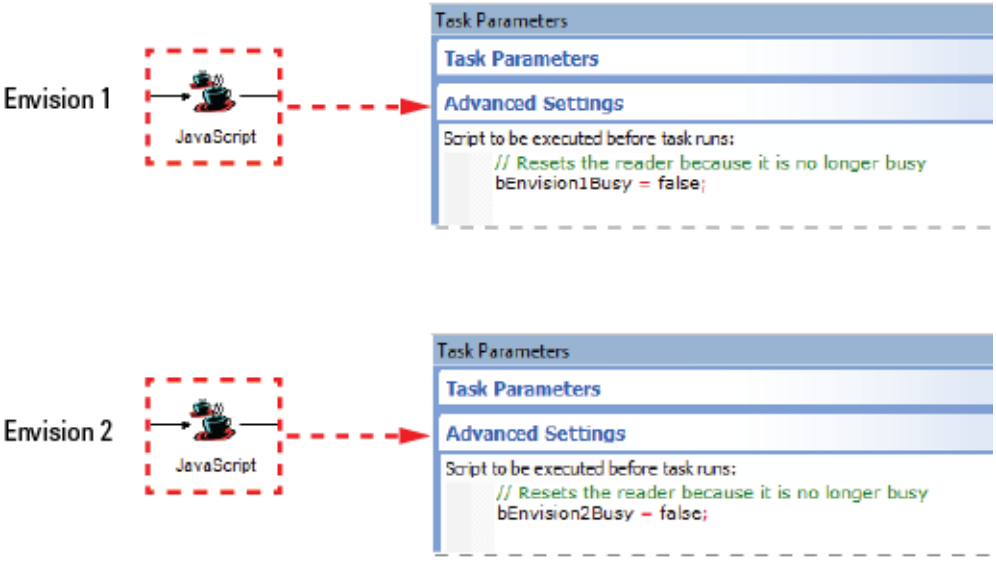


Set the **Process Control Task Parameters** as shown.

- Notice that the process selection variable is `procoption`. This variable determines which process to initiate. The value of this variable is passed from the Process Control JavaScript. If `procoption` = 1, the task will initiate option 1: Envision 1. If `procoption` = 2, the task will initiate option 2: Envision 2.
- Select **Process as subroutine**. The microplate barcode information is retained during the Envision processes.



Add the JavaScript as shown in each Envision Reader process to set the Envision Reader busy state to false. After the microplate-reading is finished, the devices are available for other labware.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67

For information about...	See...
Wait For task	"Wait For" on page 686
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

Signal

Description

The Signal task (Go Signal) permits another process that is currently in the wait state to continue to the next task. The wait state is from the Wait For task in the other process.

Note: Each Signal task can be used with multiple Wait For tasks.

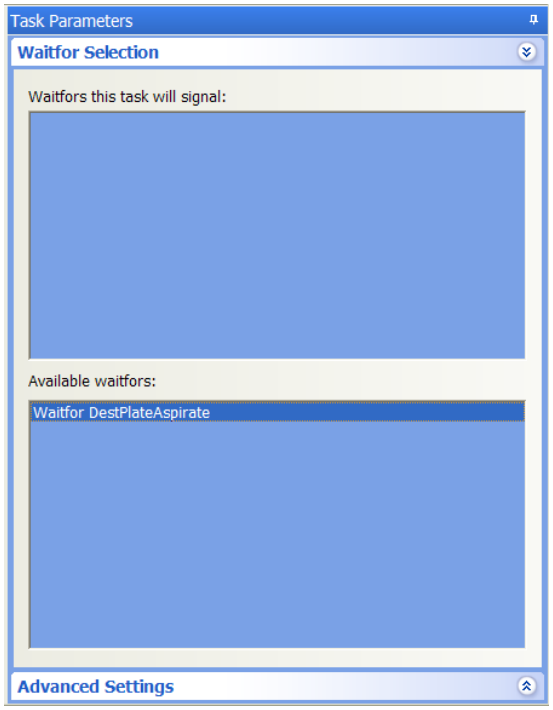
Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

Requirements

You must first add the Wait For task at the desired point in the protocol before adding the Signal task.

Waitfor Selection

After adding the Signal task at the desired point in the protocol, select the corresponding Waitfor task in the **Waitfor Selection** area. Double-click the Waitfor task name in the **Available waitfors** area. The selected Waitfor task name appears in the **Waitfors this task will signal** area.



Example

See the example in "Wait For" on page 686.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Wait For task	"Wait For" on page 686
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

Spawn Process

Description

The Spawn Process task (Spawn Process) is used to initiate another process within the same protocol. For example, to reduce evaporation, you can use Process Control task to deliver a certain labware into the system only when they are ready to be processed. You can also use the Process Control task with JavaScript code to use operator-supplied information to initiate a new process or funnel incoming labware into a different process depending on the barcode.

Task is available for...	Task is available in...
Any device	Main Protocol

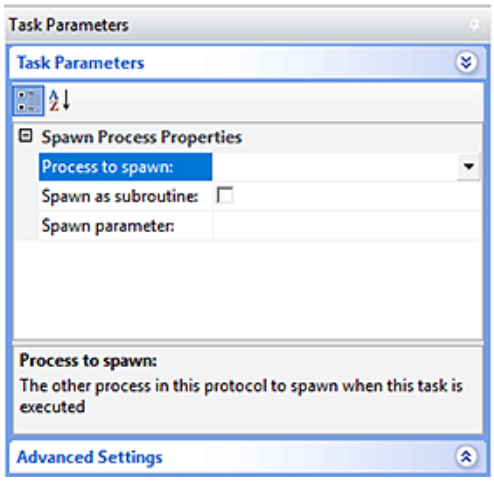
When to use Spawn Process instead of Process Control

The Spawn Process task provides a simple way to initiate another process in the same protocol. Use the task if you want to initiate only one or two processes in the protocol, and the spawning of the processes do not depend on varying run-time conditions.

If you want to initiate multiple processes depending on changing run-time conditions, use the Process Control task instead. See “Process Control” on page 666.

Task parameters

After adding the Spawn Process task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
Process to spawn	The process that starts to run when the current process reaches the Spawn Process task. Select from the list of processes.

Parameter	Description
Spawn as subroutine	The option to run the spawned process as a subroutine (or subprocess) of the current process. **IMPORTANT** If the spawned subprocess specifies a different labware type, the software ignores the labware specification. • Select the option if you want the software to start the spawned process when the Spawn Process task is reached, and continue the current process after the spawned process is finished. • Clear the check box if the two processes can run in parallel when the current process reaches the Spawn Process task. In this case, the task will use the **Spawn parameter** property if it is specified.

Spawn parameter	The option to enter a string, used by a script, to identify which process spawned a given spawned process.
	
	For example, suppose that process A and process B both contain a Spawn Process task that spawns process C. Process C has a script that behaves differently depending on whether the process is spawned by process A or B. If the Spawn Process task in process A has a blank **Spawn parameter**, but the Spawn Process task in process B has a **Spawn parameter** set to B, the script in process C could distinguish between the two as follows:
	
	`if (plate.spawnParameter == "B")`
	`{`
	`  // Process B spawned this process`
	`}`
	`else`
	`{`
	`  // Process A spawned this process`
	`}`
	
	The Spawn parameter property can be set to a scripted value or set in the task's script.

Example 1

Goal

Replicate Source Plate on the Bravo Platform. Deliver the Diluent Plate from the BenchCel Microplate Handler into the system only when the Destination Plate is ready for dilution.

Implementation

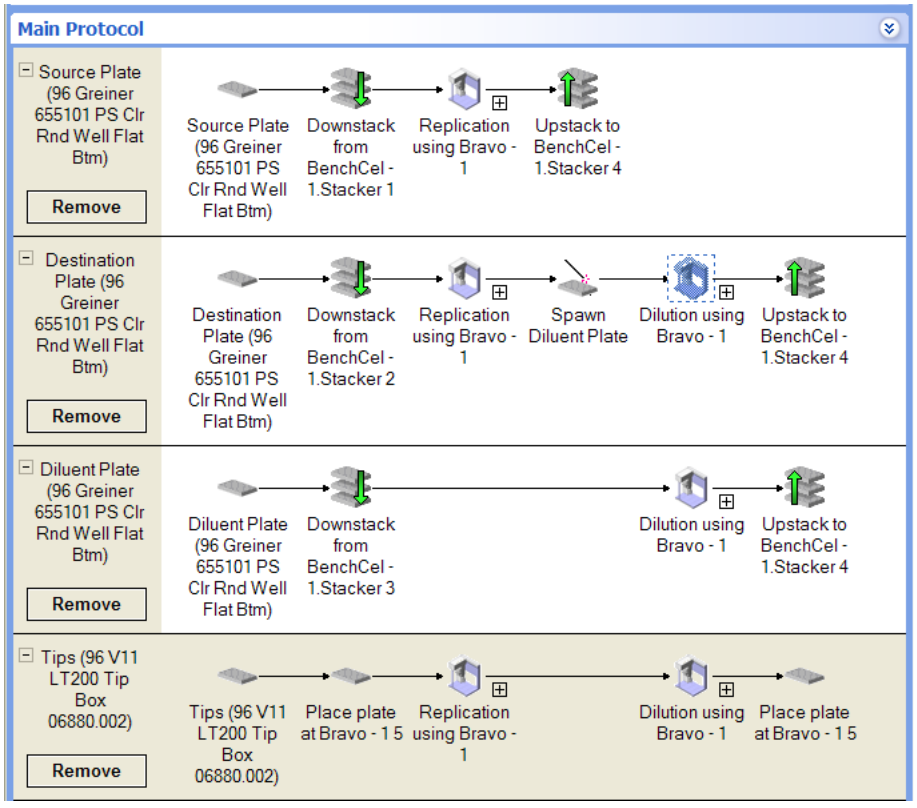
The Bravo Platform and BenchCel Microplate Handler in the example are configured as follows:

- The replication process will occur on the Bravo Platform.

- The Source Plate, Destination Plate, and Diluent Plate will be stored in different stacks in the BenchCel Microplate Handler. At the end of the process, the microplates are stored in an empty stack in the BenchCel Microplate Handler.

When writing the protocol:

- Add a process for the Source Plate as shown. In the Source Plate process, add a Bravo Subprocess (renamed Replication in the example) that aspirates contents from the Source Plate and dispenses into the Destination Plate.
- Add a process for the Destination Plate as shown. In the Destination Plate process, add two Bravo Subprocesses:
  - The first subprocess (renamed Replication in the example) reflects the movement of contents from the Source Plate and into the Destination Plate. This is a copy of the tasks in the Source Plate process.
  - The second subprocess (renamed Dilution in the example) aspirates contents from the Diluent Plate and dispenses into the Destination Plate.
- Add a process for the Diluent Plate as shown. In the Diluent Plate process, add a subprocess (renamed Diluent in the example) that reflects the movement of contents from the Diluent Plate into the Destination Plate. This is a copy of the tasks in the Diluent subprocess in the Destination Plate process.
- Add a configured labware for the Tip Box as shown.
- In the Destination Plate process, add a Spawn Process task between the Replication subprocess and the Dilution subprocess. Doing so holds the Diluent Plate in the BenchCel stack until the Replication subprocess is finished and the Destination Plate is ready to receive the diluent.



Example 2

Goal

The same goal as Example 1, except:

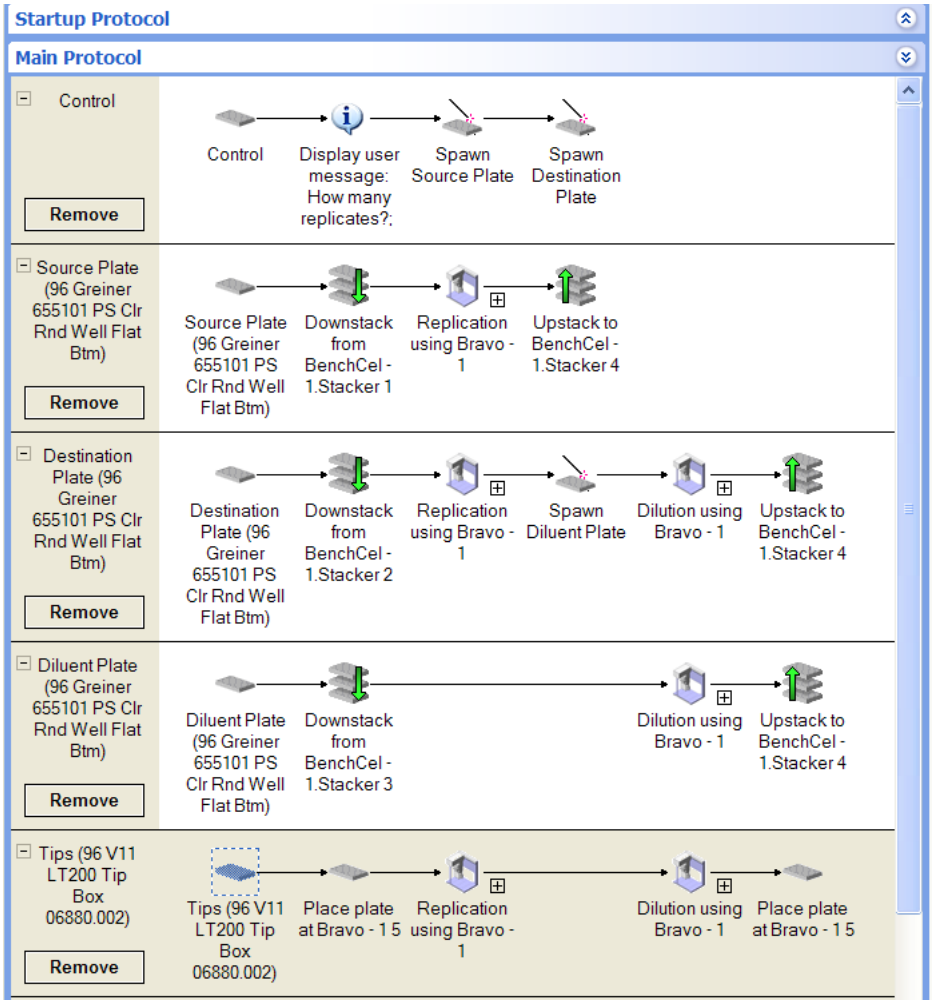
- Ask the operator to specify the number of replications.
- Control the spawning of the Source Plate and Destination Plate processes based on the operator input.

Implementation

The Bravo Platform and the BenchCel Microplate Handler are configured as described in Example 1. Set up the Source Plate, Destination Plate, and Diluent Plate processes as described in Example 1.

Add a new process called Control at the top of the Main Protocol as shown. Add the following tasks in the Control process:

- *User Message*. Asks the operator to specify the number of replications.
- *Spawn Process*. Starts the Source Plate process. In addition, provide JavaScript code in the Advanced Settings area to incorporate the input from the operator during the run. For example, if the operator specified two replications, the JavaScript code should use that value to produce two replications during the run.
- *Spawn Process*. Starts the Destination Plate process. Provide JavaScript code in the Advanced Settings area to incorporate the input from the operator during the run.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Task parameter variables	"Using simple variables" on page 93
Using JavaScript	"Using JavaScript" on page 100
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

User Message

Description

The User Message task (User Message):

- Displays reminder messages. For example, the message can remind the operator to change pipette heads, empty the waste container, empty a reservoir, or replace labware.
- Prompts operators for variable values.

The protocol run is paused until the operator clicks Continue, or supplies the requested variable value and clicks Continue.

Note: Except for those that request variable input, user messages do not appear when running a protocol in simulation mode.

IMPORTANT

Remove all user messages from protocols that will run unattended.

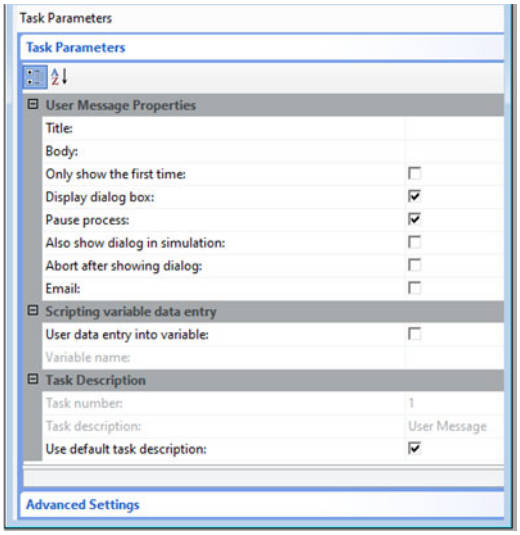
Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

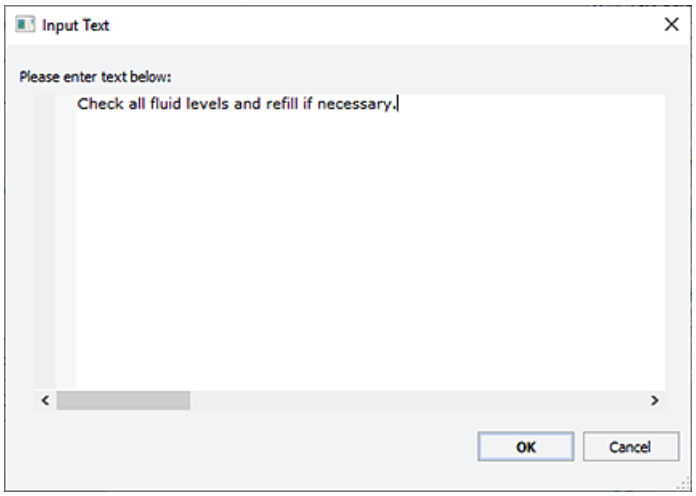
Requirements

If you want the User Message task to request variable input, add the variable in the desired tasks. For information, see "Using simple variables" on page 93.

Task parameters

After adding the User Message task at the desired point in the protocol, set the following parameters in the **Task Parameters** area:



Parameter	Description
User Message Properties	
Title	The text that appears in the titlebar of the message dialog box.
Body	The text that displays in the message dialog box. ***To enter the message using the Input Text dialog box:*** **1** Click the Body field in the Task Parameters area, and then click the ... button that appears. The Input Text dialog box opens. **2** Type the message that you want to display, and then click OK. 
Only show the first time	The option to display the message the first time it appears in the protocol if you are running the protocol multiple times, or if the User Message task is in the loop.

Parameter	Description
Display dialog box	The option to display the message dialog box when the User Message task runs.
Pause process	The option to pause the run when the User Message task runs. This option is only available when the Display Dialog Box option is selected.
Also show dialog in simulation	The option to display the User Message dialog box during a simulation run.
Abort after showing dialog	Available only if the Display dialog box check box is selected and the User data entry into variable check box is cleared. Select this option to include an Abort button in the User Message dialog box. When the Abort button is present, there is no other way to dismiss the User Message dialog box, and clicking the Abort button will abort the protocol. This option can be used to force a user to abort a protocol when an error is detected. *Note:* The Pause process option is ignored in this case. The process is always paused when the dialog is shown.
Email	The option to send an email when the User Message task runs. The email subject is the message title and the email body is the text displayed in the message dialog box. To receive email, you must set up email notification. See "Setting up email notification" on page 309 for instructions on how to set up email notification. *Note:* If the email setup is incorrect, an error will be recorded in the main log but will not cause the run to pause.
Scripting variable data entry	
User data entry into variable	The option to prompt the operator for a variable value.
Variable name	The name of the variable.
Task Description	
Task number	The number that indicates the position of the task in the protocol.
Task description	The description of the task.
Use default task description	The option to use the default task description or provide your own description for the task. Select the check box to use the default description. Clear the check box to provide your own description.

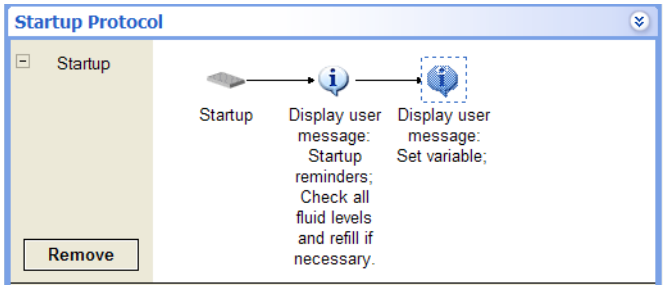
Example

Goal

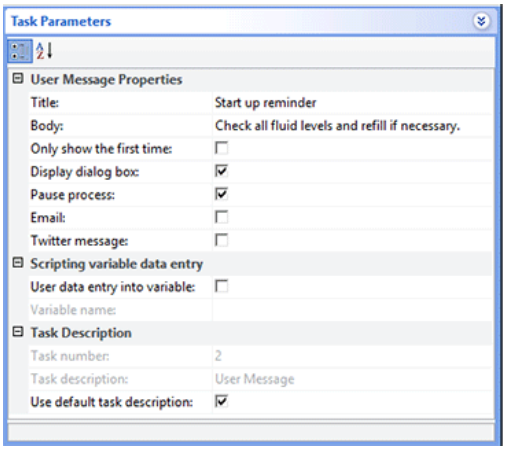
Display messages at the beginning of the protocol run to remind the operator to check fluid levels and to set the aspirate volume.

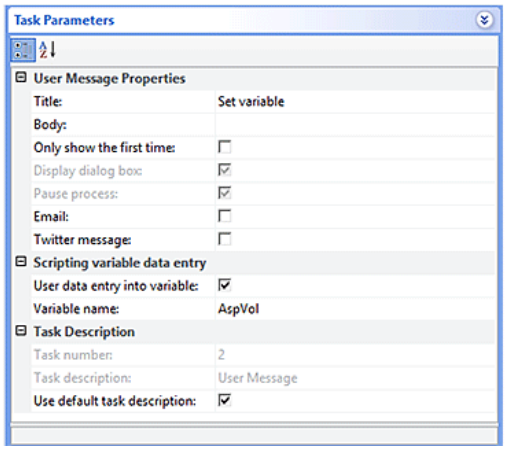
Implementation

In the Startup Protocol, add two User Message tasks. The first task asks the operator to check fluid levels. The second message asks the operator for the aspirate volume. AspVol is the variable defined for the aspirate volume.



The task parameters for both User Message tasks are as follows:





Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Using simple variables	"Using simple variables" on page 93
Adding user message prompts	"Adding user message prompts" on page 98
Defining variables	"Define Variables" on page 645

For information about...	See...
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

Wait For

Description

In a multi-process protocol, the Wait For task (Wait For) pauses the process that contains the task and waits for the go-ahead signal from another process before continuing to the next task. The go-ahead signal comes from the Signal task in the other process.

Note: Multiple Wait For tasks can be used with a single Signal task.

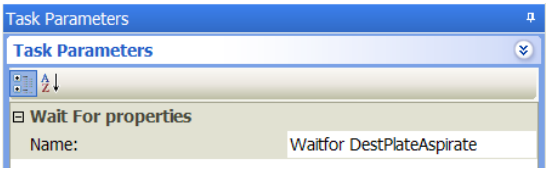
Task is available for...	Task is available in...
Any device	Startup Protocol Main Protocol Cleanup Protocol

Requirements

The Wait For task is always used with the Signal task. You must first add the Wait For task at the desired point in the protocol before adding the Signal task.

Task parameters

After adding the Wait For task at the desired point in the protocol, double-click the corresponding Wait For task in the **Task Parameters** area:



Parameter	Description
Name	The name you want to assign the Wait For task. Type a name that describes the waitfor condition.

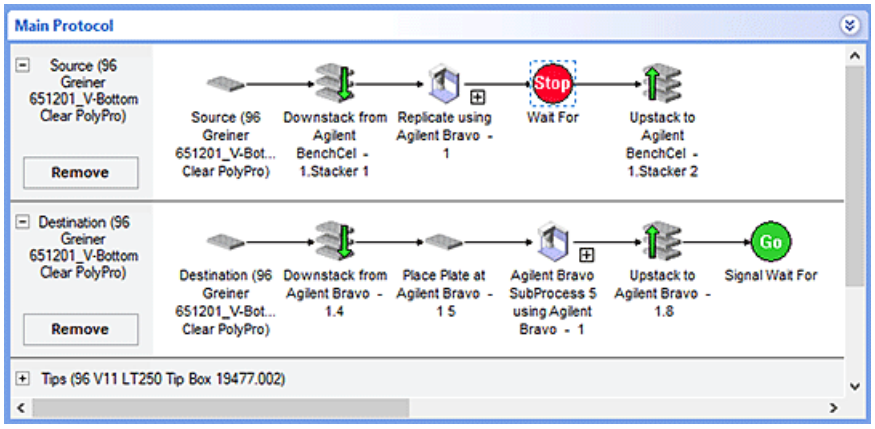
Example: Finish all liquid-handling tasks before incubation

Goal

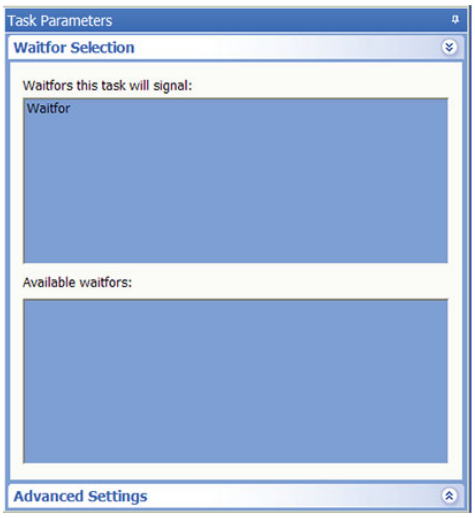
Make sure all liquid-handling tasks are finished before storing the source microplates.

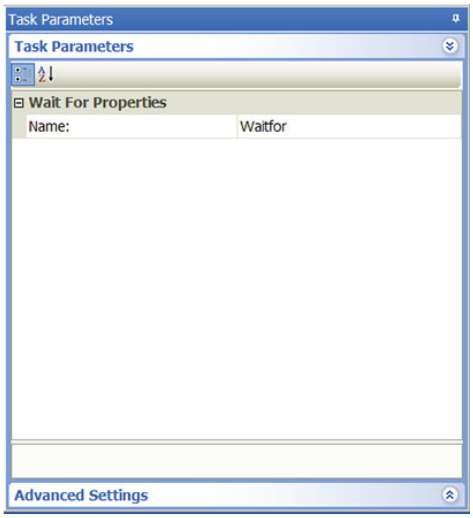
Implementation

In the protocol where the source microplate and destination microplates are separate processes, add a Wait For task in the source microplate process, after the liquid-handling tasks. Add a Signal task in the destination microplate process after the liquid-handling tasks.



The Wait For and Signal task parameters are set up as follows:





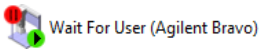
For other examples, see "Dismount" on page 367 and "Mount" on page 374.

Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
Signal for task	"Signal" on page 675
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

Wait for User (Agilent Bravo)

Description

The Wait For User (Agilent Bravo) task () pauses the protocol and waits until the operator presses the go button on the pendant. When the operator presses the go button, the protocol resumes.

You use the Wait For User task if you want to perform a task manually at the desired point in the protocol. For example, you can use the Wait For User task to pause the run so that you can manually replace a microplate on the deck. When you are finished, you press the go button on the pendant.

Note: Alternatively, you can use the User Message task to insert a pause in a protocol. Use the User Message task instead of the Wait For User task if you have easier access to the computer than the pendant during a run.

Task is available for...	Task is available in...
Bravo Platform that is equipped with a legacy pendant that has a Go button	Main Protocol

Task parameters

The Wait For User task does not have task parameters.

Example

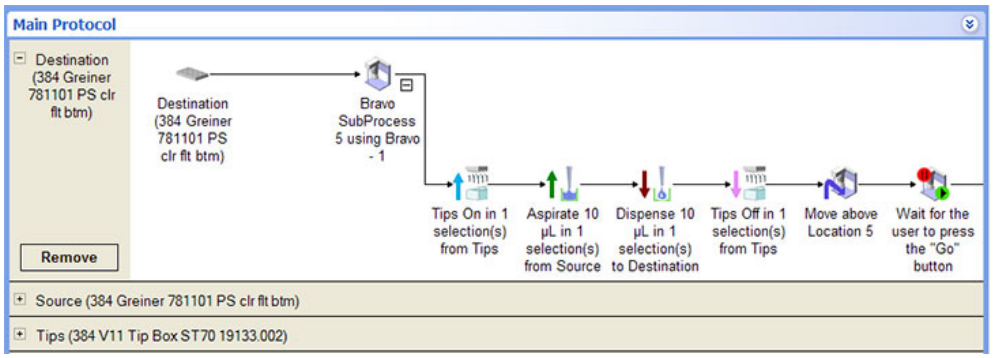
Goal

After the first dispense task, move the pipette head out of the way. Pause the run so that you can replace the destination microplate on the Bravo deck. Press the Go button on the pendant after the new destination microplate is placed.

Implementation

A Move to Location task is added after the first Dispense task to move the pipette head away from the current destination microplate. In the following example, because the destination microplate is at location 1, the pipette head is moved to location 5.

A Wait for User task is added after the Move to Location task to pause the run.



Related information

For information about...	See...
Adding devices	• "Adding devices" on page 26 • Device user guide
Adding tasks in a protocol	"Adding and deleting tasks" on page 67
User Message task	"User Message" on page 682
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449
Microplate-handling tasks	"Setting parameters for microplate-handling tasks" on page 343
Microplate-storage tasks	"Setting parameters for microplate storage tasks" on page 411

11 Specifying pipetting techniques

This chapter contains the following topics:

- "About the Pipette Technique Editor" on page 692
- "Creating and editing pipetting techniques" on page 693
- "Managing pipetting techniques" on page 697

About the Pipette Technique Editor

Pipette techniques

You can define a pipette technique to use different pipetting methods. Different applications can benefit from different pipetting methods. For example, in multiplexed microplates, dispensing at an offset from the well center can improve distribution of the fluid. In cell-based assays, moving the pipette to the side of the wells minimizes the removal of the cells from the center or reading area of the well.

Pipette Technique Editor

You use the Pipette Technique Editor to define any number of pipette techniques. After you create a technique, it becomes available for the following liquid-handling tasks in the Bravo Subprocess:

- Aspirate
- Dispense
- Mix
- Pin Tool

Related information

For information about...	See...
Creating and editing Pipette Techniques	"Creating and editing pipetting techniques" on page 693
Managing Pipette Techniques	"Managing pipetting techniques" on page 697
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

Creating and editing pipetting techniques

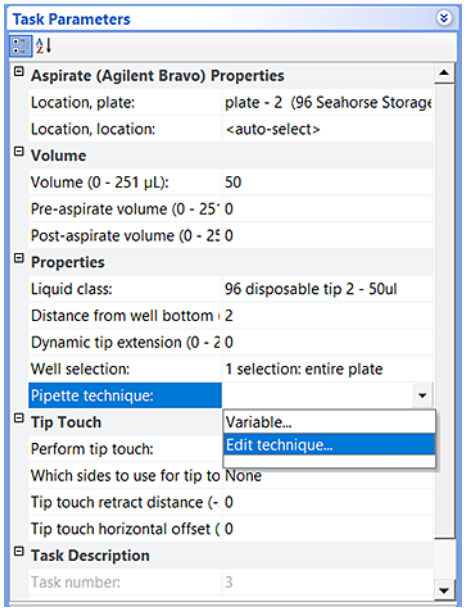
You can create a pipette technique in two ways:

- When setting task parameters in a protocol
- Using the Tools menu

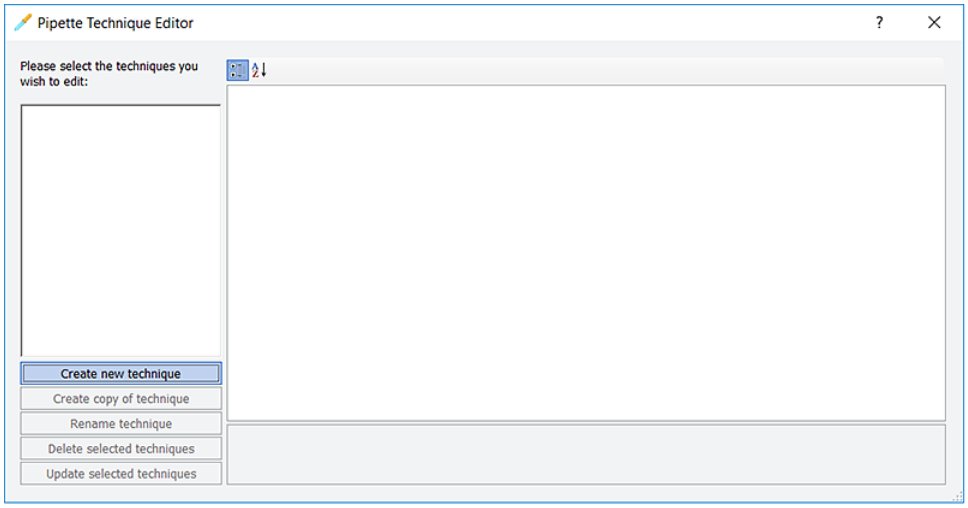
Creating a pipette technique when setting task parameters

To create a pipette technique when setting task parameters:

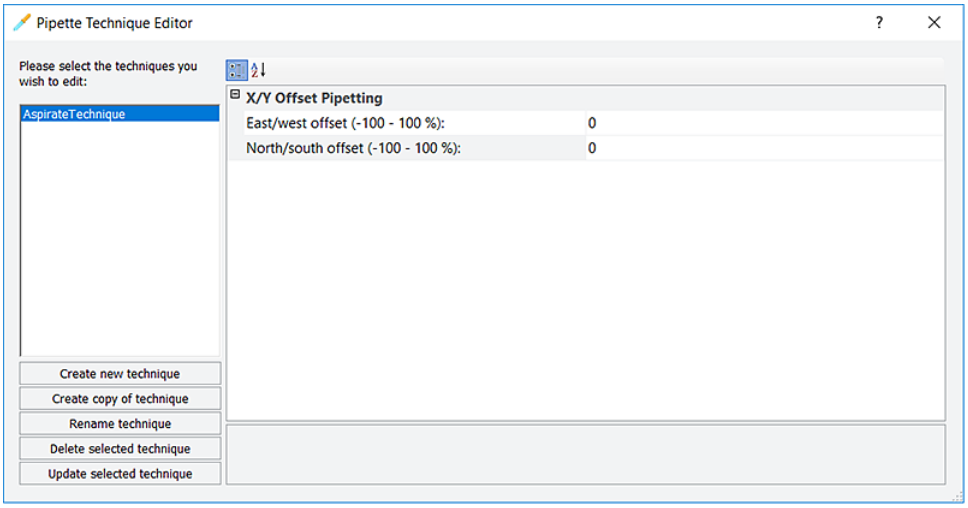
1. In the VWorks window, create a new protocol.
2. Add the **Subprocess (Agilent Bravo)** task.
3. Add one of the following tasks:
   - Aspirate
   - Dispense
   - Mix
   - Pin Tool
4. In the **Task Parameters** area, select **Pipette technique > Edit technique**.



The Pipette Technique Editor dialog box opens.



5 Click **Create new technique**, type a technique name, and then click **OK.** The new technique name appears in the Pipette Technique Editor dialog box. In addition, the **X/Y Offset Pipetting** parameters appear to the right of the technique name.

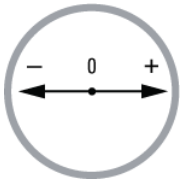


6 *VWorks Plus only.* If an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

7 In the **East/west offset (-100 - 100%)** box, type the distance (in percent of well radius) for the pipette to move in the X direction:

- 0 does not move the pipette horizontally.
- Positive value moves the pipette to the right.
- Negative value moves the pipette to the left.

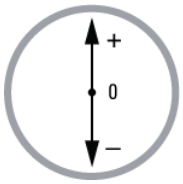
Note: The pipette offset directions do not correspond to the robot jog directions in Bravo Diagnostics.



8 In the **North/south offset (-100 - 100%)** box, type the distance (in percent of well radius) for the pipette to move in the Y direction.

- 0 does not move the pipette forward or backward.
- Positive value moves the pipette backward.
- Negative value moves the pipette forward.

Note: The pipette offset directions do not correspond to the robot jog directions in Bravo Diagnostics.



9 When you are finished, click **Update selected technique**.

10 *VWorks Plus only.* If an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

Creating a pipette technique using the Tools menu

To create a pipette technique using the Tools menu:

1 In the VWorks window, click **Tools > Pipette Technique Editor.** The Pipette Technique Editor dialog box opens.

2 Click **Create new technique,** type a technique name, and then click **OK.** The new technique name appears in the Pipette Technique Editor dialog box. In addition, the X/Y Offset parameters appear to the right of the technique name.

3 *VWorks Plus only.* If an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

4 Set the parameters in the **X/Y Offset Pipetting** table.

5 When you are finished, click **Update selected technique**.

6 *VWorks Plus only.* If an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

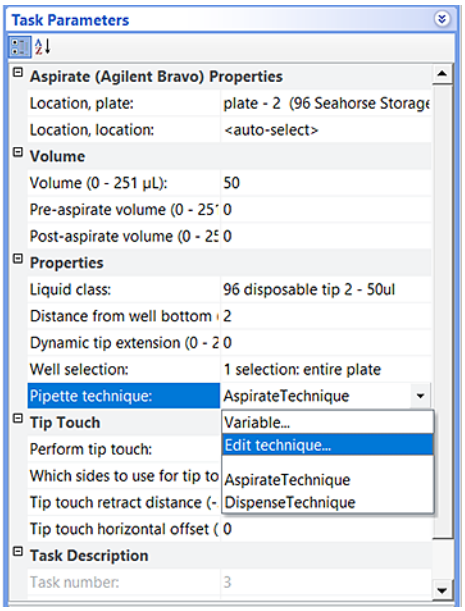
Editing a pipette technique when setting task parameters

To edit a Pipette Technique when setting task parameters:

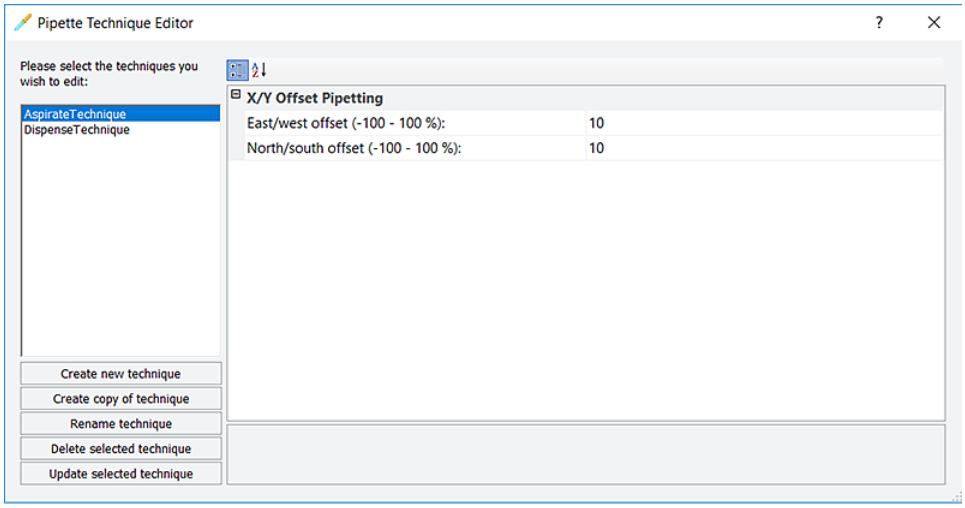
1 In the VWorks window, open the protocol.

2 In the **Main Protocol** area, select the task for which the pipette technique will change.

3 In the **Task Parameters** area, select **Pipette technique > Edit technique**.



The Pipette Technique Editor dialog box opens.



4 Select the pipette technique on the left side of the dialog box.

5 Make the desired changes in the **X/Y Offset Pipetting** table.

6 When you are finished, click **Update selected technique**.

7 *VWorks Plus only.* If an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

Editing a pipette technique using the Tools menu

To edit a Pipette Technique using the Tools menu:

1 In the **VWorks** window, click **Tools > Pipette Technique Editor.** The Pipette Technique Editor dialog box opens.

2 Select the pipette technique on the left side of the dialog box.

3 Make the desired changes in the **X/Y Offset Pipetting** table.

4 When you are finished, click **Update selected technique**.

5 *VWorks Plus only.* If an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

Pipette technique storage

Shared Services saves each pipette technique as an XML file in a VWorks Projects folder, for example, /VWorks Projects/VWorks/Pipetting/Techniques.

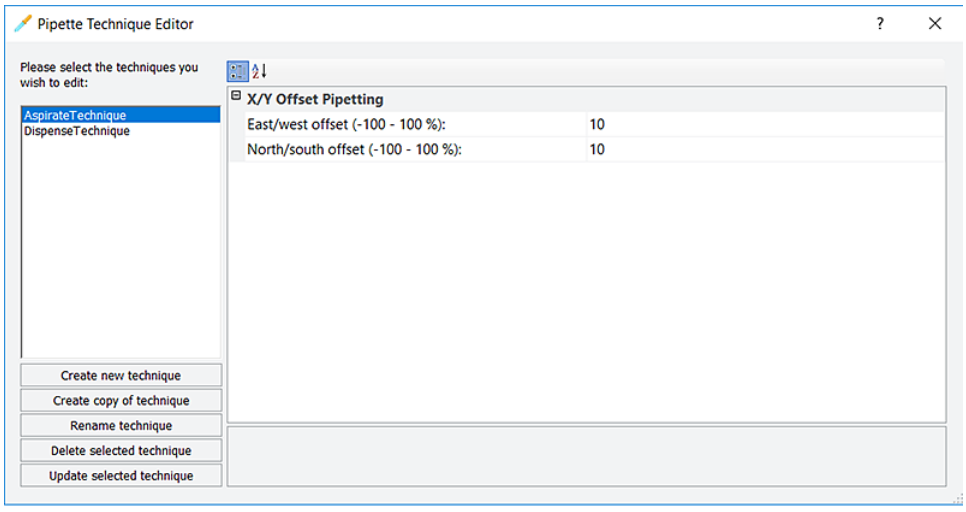
Related information

For information about...	See...
Pipette Techniques	"About the Pipette Technique Editor" on page 692
Managing Pipette Techniques	"Managing pipetting techniques" on page 697
Liquid-handling tasks	"Setting parameters for liquid-handling tasks" on page 449

Managing pipetting techniques

To copy, rename, and delete an existing Pipette Technique:

1 In the VWorks window, click **Tools > Pipette Technique Editor.** The Pipette Technique Editor dialog box opens.



2 Select the Pipette Technique on the left side of the dialog box.

3 Click one of the following:

- **Create copy of technique**. The software prompts you to type a new name for the duplicated technique, and then creates a copy of the selected technique and saves it using the new name.
- **Rename technique**. A message warns you that the change will affect all protocols that use the technique. If you click Yes to confirm the change, the software prompts you to type a new name for the selected technique, and saves the technique using the new name.
- **Delete selected technique**. A message warns you that the change will affect all protocols that use the technique. If you click Yes to confirm the change, the software deletes the selected technique.

4 *VWorks Plus only.* If an audit trail is being logged, the **Audit Comment** dialog box opens. Select or type a reason for the change, and then click **OK**.

Related information

For information about...	See...
Pipette Techniques	"About the Pipette Technique Editor" on page 692
Creating and editing Pipette Techniques	"Creating and editing pipetting techniques" on page 693

12 Maintenance and troubleshooting

This chapter contains the following topics:

- "Backing up and restoring VWorks files" on page 700
- "Exporting and importing protocols and associated records" on page 703
- "Viewing logs" on page 714
- "Resolving device initialization errors" on page 721
- "Resolving compilation error messages" on page 722
- "Disabling and enabling tasks" on page 723
- "Using breakpoints to monitor and troubleshoot tasks" on page 724
- "Resolving barcode reader error messages" on page 729
- "Recovering from deadlocks" on page 730
- "Setting up automated error responses" on page 735
- "Reporting problems" on page 743

Backing up and restoring VWorks files

You should regularly back up data and files in case they become damaged or lost. The backup copy should be stored on a different computer or storage device. See the Administrator Guide for your VWorks edition for more details on disaster recovery plans and data and software backup procedures.

Types of backups

The following table lists the available data backup and restore features.

Feature	Description	Role	See...
VWorks Plus only. Procedures for total system backup and restore	Backs up and restores all the VWorks records and data, including the Secure Storage Repository and Shared Services database, which contains data from the Control Panel, such as user management data and licenses.	System administrator	"Configuration and administration guides" on page 3*VWorks Plus Configuration and Administration Guide*
VWorks Standard only. Procedures for total system backup and restore Shared Services Maintenance Utility	Backs up and restores all the VWorks records and data, including the Shared Services database, which contains data from the Control Panel, such as user management data and licenses.	VWorks Administrator	"Guides for VWorks Standard edition" on page 3*VWorks Standard Configuration and Administration Guide*
VWorks Backup Manager > Backup	Backs up only the following: • Existing state of the inventory database • Labware definitions and liquid classes • Pipette techniques Stores the backup in a .vbk file.	VWorks Administrator or technician	Procedures in this topic
VWorks Backup Manager > Restore	Restores backed up records from a .vbk file.	VWorks Administrator	Procedures in this topic
VWorks Export VZP File	Enables you to make a copy of selected protocols and associated files, which you can transfer to another computer.	VWorks Administrator or technician	"Exporting and importing protocols and associated records" on page 703
VWorks Import VZP File	Enables you to import an exported VWorks VZP file.	VWorks Administrator	"Exporting and importing protocols and associated records" on page 703
Experiments Editor archive and export	Provide files in XML format that can be used to backup the Experiments database.	VWorks Administrator or technician	"Exporting and importing experiment IDs" on page 77

Using the Backup Manager

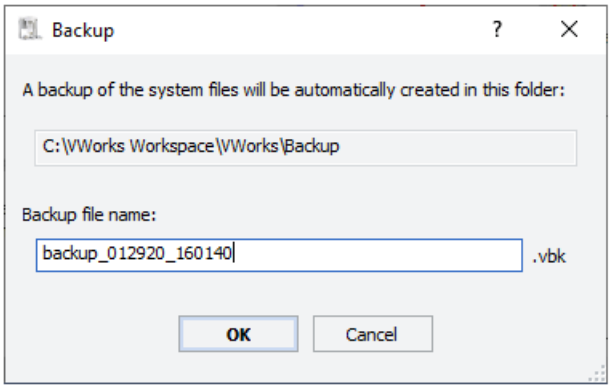
You use the Backup Manager to back up the inventory databases, labware definitions and liquid classes, and pipette techniques. The Backup Manager stores the backup in a .vbk file.

Make sure the inventory database is up-to-date before you make a backup copy.

If you want to back up the inventory database without backing up the entire set of system files, see the *VWorks Automation Control Setup Guide*.

To use Backup Manager to make a backup copy:

1. In the **VWorks** window, select **Tools > Backup Manager > Backup.** The Backup dialog box opens.



2. Type a name for the backup file.
3. Click **OK.** VWorks stores the backup copy in a .vbk file in the following location: \VWorks Workspace\VWorks\Backup folder.

To use Backup Manager to restore a backup copy:

1. In the **VWorks** window, select **Tools > Backup Manager > Restore.** The Open dialog box appears.
2. Locate and select the .vbk file, and then click **Open.**
3. When the confirmation message appears, do one of the following:
   - Click **No** to cancel the restore process.
   - Click **Yes** to overwrite existing labware definition, liquid classes, and pipette techniques.

     VWorks Plus only. In addition, the software does the following:
     - Performs tamper detection on the backup file (.vbk) before parsing the file contents. If the software detects that the file is corrupted or has been modified outside of VWorks 14.0 or later, an error message appears and you cannot restore the file.
     - Prevents you from overwriting a file of the same name if the file is in the validation or released state.

Related information

For information about...	See...
Exporting and importing protocols and associated files	"Exporting and importing protocols and associated records" on page 703
VWorks software components: protocol, device file, and so on	"Relationship of components in VWorks software" on page 5
Viewing logs	"Viewing logs" on page 714
Contacting Agilent Technologies and Reporting problems	"Reporting problems" on page 743

Exporting and importing protocols and associated records

Reasons for exporting and importing protocol files

You can export and import protocols and their associated records to transport protocols between computers, back up and recover protocols and associated files, and to facilitate troubleshooting problems with protocols and other files when seeking assistance from Agilent Technical Support.

VWorks files that you can export

You can export or import the following:

- Protocol file
- Form file
- Runset file
- Device file, including the device profile and teachpoint file
- Macros used in the exported protocol or the entire macro library (.mlb file)
- Labware definitions
- Liquid classes
- Pipette techniques
- Hit-picking files
- Plate map files
- Barcode files
- Error library
- JSWrapper task files
- Log files

Exporting protocols and associated records

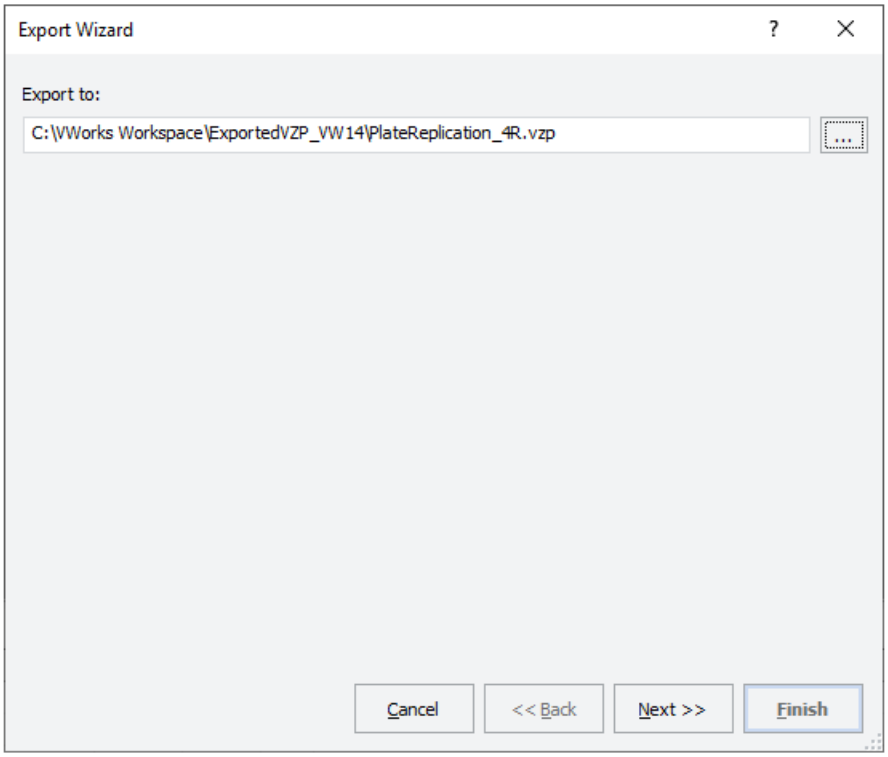
Before you start

You can export files from Shared Services storage if you have administrator, technician, or operator privileges.

VWorks Plus only. The software performs tamper detection on the files selected for export and prevents you from exporting a file that is corrupted or has been modified outside of the VWorks software.

To export the files:

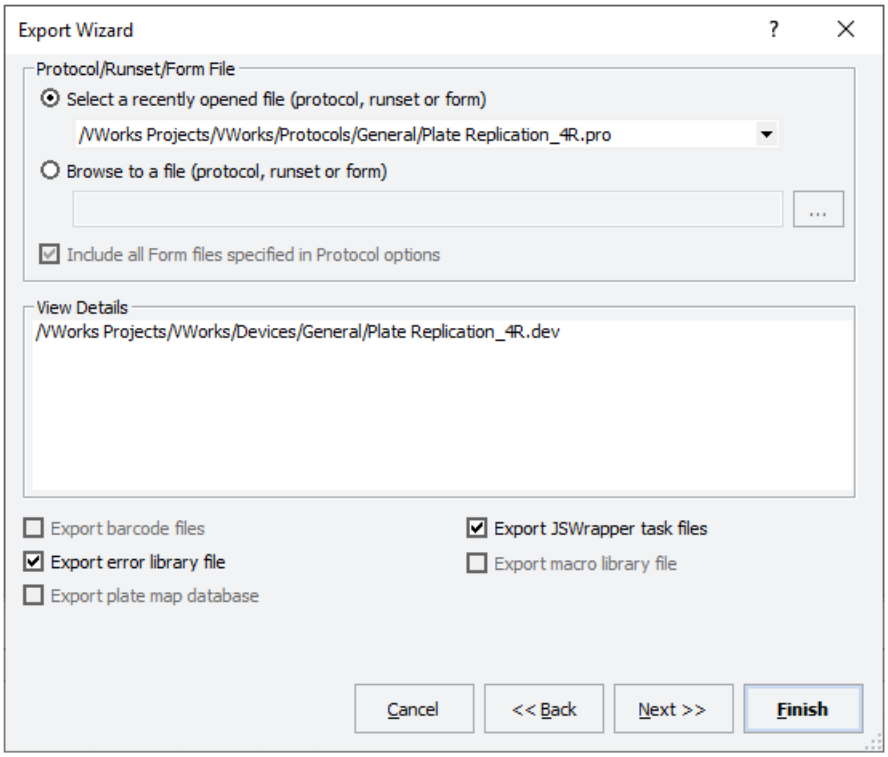
1 In the **VWorks** window, select **File > Export VZP File.** The Export Wizard opens.



2 Click (...).

In the **Save As** dialog box that opens, select the location and type a name for the .vzp file that you want to export.

3 In the **Export Wizard**, click **Next**. The Protocol/Runset/Form File page opens.



4 Follow the instructions in the wizard to select the following:

IMPORTANT

Any custom image files in a form must be in the same folder as the .VWForm file that you are exporting.

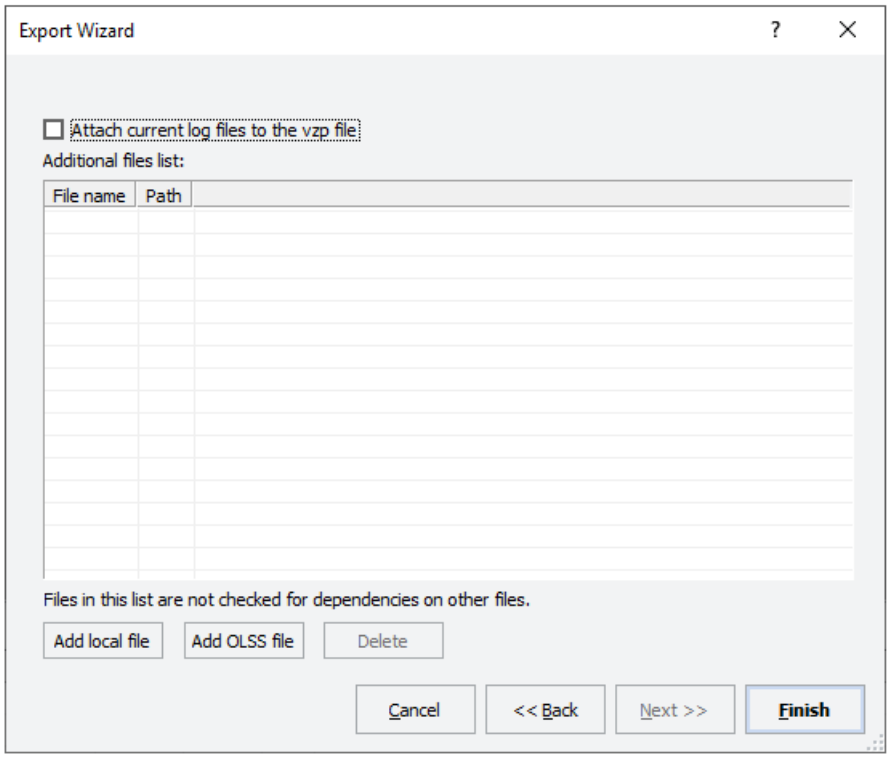
VWorks Plus only. If the selected file or any of its dependent files has been corrupted or modified outside of the VWorks software an error message appears and the file cannot be loaded for importing.

Property in wizard	Description
Protocol/Runset/Form file	Select the file that you want to export. You do not need to select the associated device file and other files. The software automatically exports all the associated files.
Include all Form files specified in Protocol options	*Optional*. If a form (.VWForm file) is specified in the protocol's options, verify the this check box is selected (default) to include the form in the export. If you do not want to include the form, clear the check box.
Export barcode files	*Optional*. If the protocol is associated with barcode files, select the check box to include the files in the exported and compressed files.
Export error library file	*Optional*. If the protocol is associated with an error library file, select the check box to include the file in the exported and compressed files.
Export plate map database	*Optional*. If the protocol is associated with a plate map database, select the check box to include the database in the exported and compressed files.

Property in wizard	Description
Export JSWrapper task files	*Optional*. If the protocol includes JavaScript Wrapper tasks, select the check box to include the files in the exported and compressed files.
Export macro library file	*Optional*. If the protocol contains task macros that have the same names as macros in the Available Macros area in the VWorks window (macro library), do one of the following: • Select the **Export macro library file** check box to export the macro library information for those macros from the protocol that are also in the macro library. After the .vzp file is imported, this set of macros can be added to the macro library on the destination computer. • Clear the **Export macro library file** check box to export the protocol version of the macros only. After the .vzp file is imported, the macros will appear in the protocol only. *Note:* The Export macro library file check box is available only for the macros in the protocol that are also in the macro library. If you want to export the entire macro library, you can attach the .mlb file on the Additional files list page of the Export Wizard.

5 Click **Next**, and then select any additional files to be included in the exported .vzp file. You can choose from the following:

- **Attach current log files to the vzp file**. Select the check box if you want to include the log files for the current VWorks session.
- **Add local file**. Opens the Open File dialog box for the local file system, for example, C:\\VWorks Workspace\VWorks.
- **Add OLSS file**. Opens the Open File dialog box for the Shared Services storage location, for example, /VWorks Projects/VWorks.



6 When you are finished, click **Finish.** The .vzp file appears in the specified location.

You can move the .vzp file to another computer and import it into the VWorks software, or send the .vzp file to Agilent Technical Support to report a problem.

Importing protocols and associated records

Before you start

You can import a VWorks .vzp file if you have administrator privileges.

IMPORTANT

Close any open protocols before you import a protocol.

VWorks Plus only. If the imported .vzp file was exported from VWorks 14.0 or later, the software does the following:

- Performs tamper detection on the imported file before parsing the file contents. If the software detects that the file is corrupted or has been modified outside of VWorks 14.0 or later, an error message appears and you cannot import the file.
- Prevents you from overwriting a protocol file of the same name if the file is in the validation or released state. In this case, you can import the file only if you rename it.

IMPORTANT

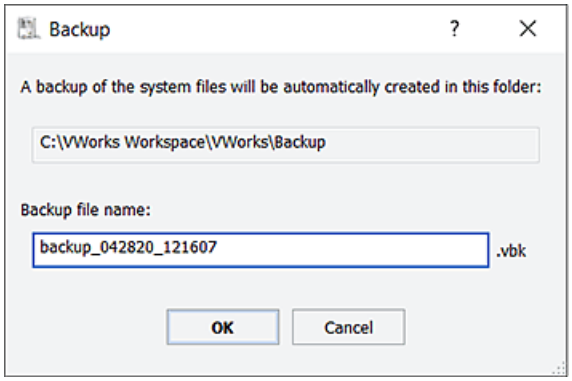
When you import a .vzp file, audit trails of the imported records are also imported. However, each imported record's version number will start over at 1. Earlier entries in each record's audit trail will use the version number of record on the computer that exported the .vzp file. Therefore, subsequent changes to an imported record may have lower version numbers than earlier entries in the record's audit trail.

To import the system files:

1 In the **VWorks** window, select **File > Import VZP File.**

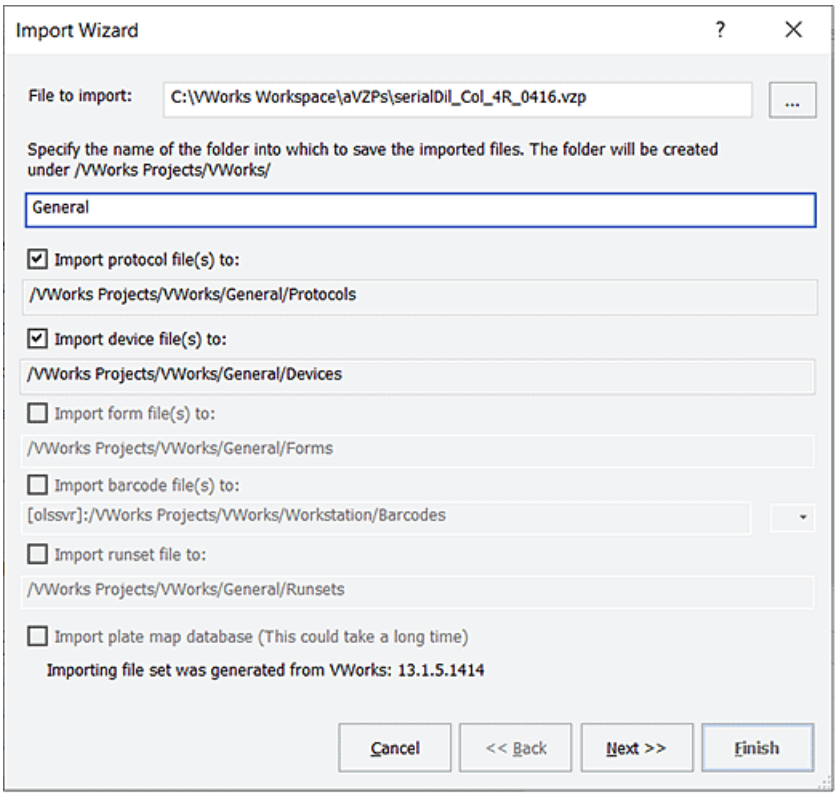
2 When a message appears and asks if you want to back up the existing labware definitions, liquid classes, pipette techniques, and plate-map databases, do one of the following:

- Click **Yes** to back up the information. In the **Backup** dialog box, type a name for the backup file (.vbk), and then click **OK.** The .vbk file is created and stored in the …\VWorks Workspace\VWorks\backup folder.



- Click **No** to start the import process.

The Import Wizard opens.



3 In the **Import Wizard**, do the following:

a Specify the **File to import**: Click ..., and then select the .vzp file in the **Open** dialog box. The selected file path appears in the **File to import** box.

Note: The .vzp file contains the protocol and associated files.

b Specify the name of the destination folder for the imported files. You can use the default name (General) or type a new name.

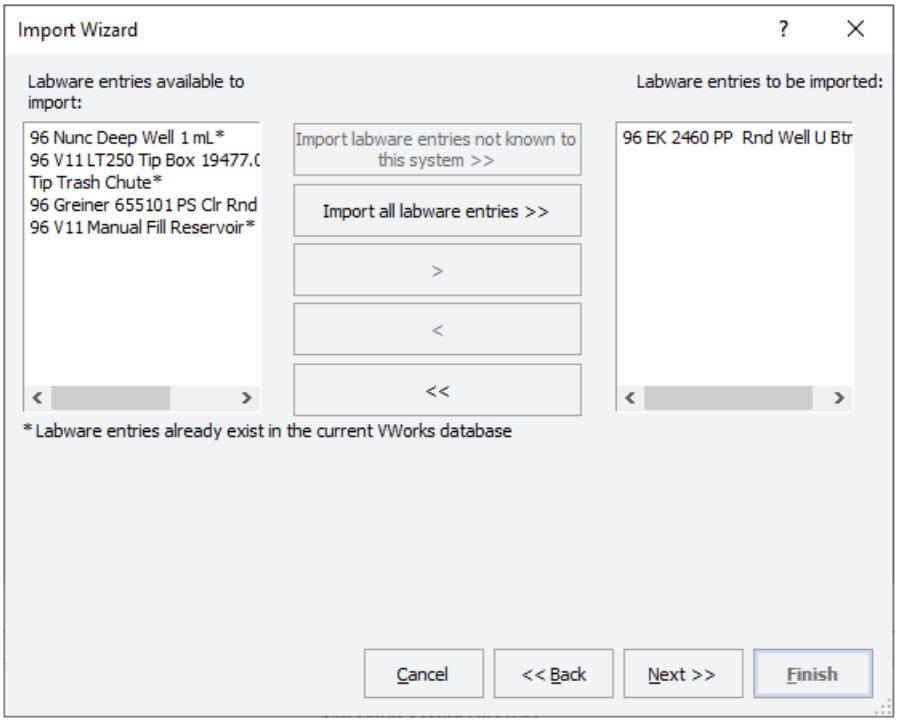
c Select the check boxes of the files types to be imported.

d If you select the **Import barcode file(s) to** check box, click the ▾ and select one of the following:

- **Browse local storage**. Allows you to select a file only from the local file system, for example, C:\VWorks Workspace\Barcode Input Files.
- **Browse Shared Services storage**. Allows you to select a file only from Shared Services storage, for example, /VWorks Projects/VWorks/Workstation/Barcodes.

e Click **Next.**

4 Follow the instructions in the wizard to select the labware entries and classes you want to import.



An asterisk highlights any files that are already in the system.

If a labware entry or class has the same name as one in the database, you have the following options:

- Replace existing labware entries or classes with newly imported entries and classes.
- *Labware classes only*. Append new labware entries to the existing labware class.
- Create a new file for the newly imported labware entries and classes. You can choose to append the import date on the name of either the new labware file or the existing labware file. The imported protocol will use the imported labware definitions. When running existing protocols, you have to use the old labware file.

CAUTION **If you replace pre-existing labware entries with new entries of the same name, the protocols that rely on existing labware entries or classes might be affected.**

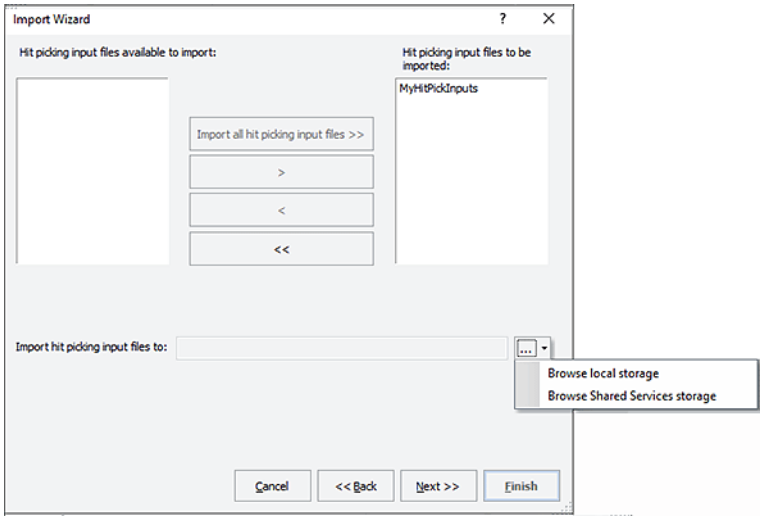
VWorks Plus only. If a labware entry or class has the same name as one in the database that is in the validation or released state:

- *Labware entry*. You may import the file with the date appended to the names of the newly imported labware entries, but you cannot overwrite the labware entry.
- *Labware class*. You cannot overwrite a labware class that contains labware entries, which are in the validation or released state. You may import the labware class if the date is appended to the name of the imported labware class.

5 Click **Next**, and follow the instructions in the wizard to select the remaining items to be imported.

For each file type, an asterisk highlights any items of the same name that are already in the system. The following table lists the potential file types. The wizard presents a page with import options for each file type associated with the File to import that you selected in step 3 of this procedure.

Note: For VWorks Plus, the available options can vary for record file of the same name that is already in the system, depending on the record's state.

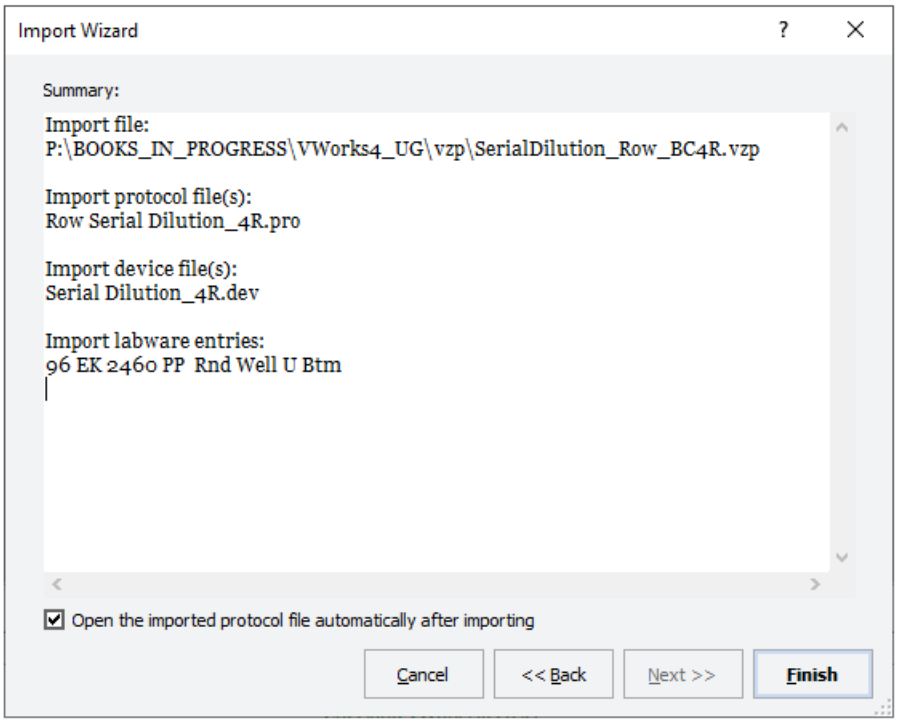
File to import	Action
Pipette technique files	Select the pipette technique files.
Hit-picking input and format files	**1** Select the hit-picking format and input files. **2** For the hit-picking input files, click the ▾ for **Import hit picking input files to**, and then click on the following: • **Browse local storage**. Allows you to select a file only from the local file system, for example, C:\VWorks Workspace\VWorks\Hit Picking. • Browse **Shared Services storage**. Allows you to select a file only from Shared Services storage, for example, [olssvr]:/VWorks Projects/VWorks/Workstation/Hitpick Input Files. 
Device profiles	Select the profiles.
JSWrapper task files	Select the JS Wrapper task files.
Error handlers	**1** Select either the error handlers not known to the system or all error handlers. **2** Specify one of the following: • Append the error handlers to the existing error library file. • Import error library file (.elb) and replace the existing one. For this option, specify the storage location.

File to import	Action
Macros	You can choose to overwrite the macros of the same name or append the date on the name of the imported macros. 1 Select either those not known to the system or all macros. 2 Specify one of the following: • Append the macros to the existing macro library file. • Import macro library file (.mlb) and replace the existing one. For this option, specify the storage location.

VWorks Plus only. You can import the Audit Comment dialog and audit trails for the error and macro libraries.

3 When you are finished, review the **Summary** page of the Import Wizard, and then click **Finish.** The files are imported in the specified locations.

Note: If you selected the option, the imported protocol opens in the VWorks window.



Related information

For information about...	See...
Contacting Agilent Technologies and reporting problems	"Reporting problems" on page 743
VWorks software components: protocol, device file, and so on	"Relationship of components in VWorks software" on page 5
Viewing logs	"Viewing logs" on page 714

For information about...	See...
Backing up VWorks files	"Backing up and restoring VWorks files" on page 700
VWorks Plus. Full system backup and restore procedures	*VWorks Plus Configuration and Administration Guide*
VWorks Standard. Full system backup and restore procedures	*VWorks Standard Configuration and Administration Guide*
Macros in protocols	"Using macros to create protocols" on page 265
Protocol forms	"Creating forms for protocol operators" on page 177

Viewing logs

About the logs

The VWorks software records events that occur and stores the information in the following logs:

- *Main Log.* Contains all of the actions that occur in the software, including changes to device profiles in the diagnostics software. For details on the log contents, see "Main Log contents" on page 718.
- *Pipette Log.* Contains all pipetting events and error information. For details on the log contents, see "Pipette Log contents" on page 719.
- *Time Constraints Log*. Contains all information about time-limited tasks. For details on the log contents, see "Time Constraints Log contents" on page 719.

The log text colors can help you distinguish between different types of messages and events.

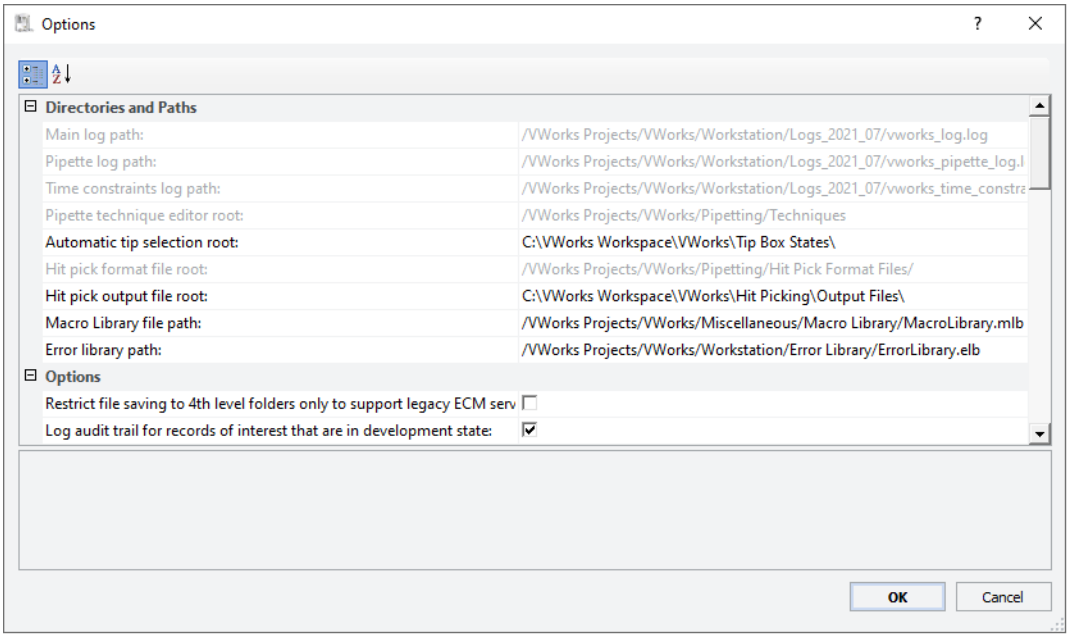
You cannot edit or delete log entries within the VWorks software, and you cannot change the storage location for the log files. However, you can download copies of the log files from Shared Services storage to a location of your choice.

Log file locations

To display the log file locations:

In the **VWorks** window, click **Tools > Options.** The Options dialog box opens.

Under **Directories and Paths**, notice that the Main log path, Pipette log path, and Time contraints log path are grayed, indicating that these paths cannot be changed.



Viewing the logs

You can view the VWorks logs in the following ways:

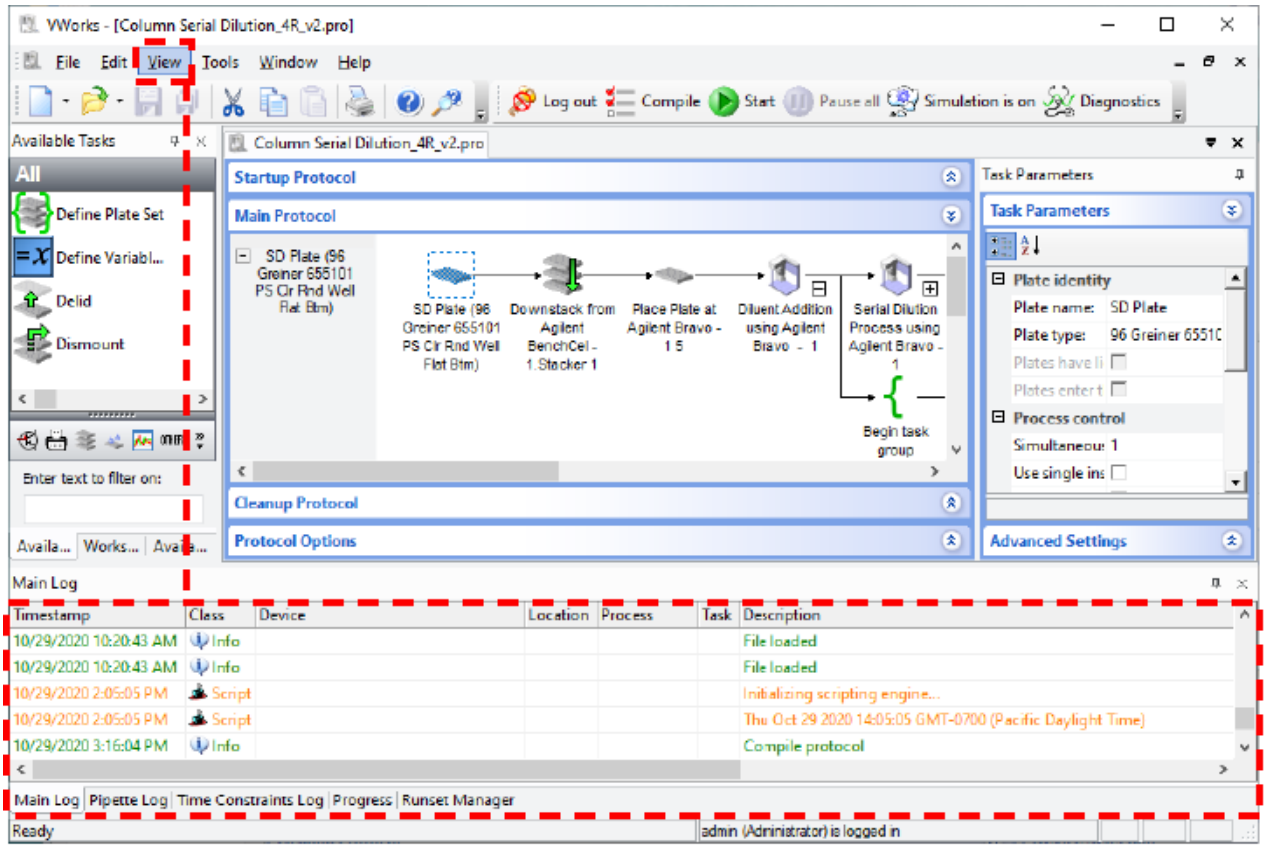
- *VWorks window*. You can display the logs for the current VWorks session in the VWorks window.
- *VWorks log files*. You can display the contents of log files from previous VWorks sessions in the following ways:
  - *View a downloaded copy of the log file*. If you are a VWorks administrator or technician, you can use the Log Management command to save a copy of a log file to a location of your choice. You can use a text editor, such as Notepad, to view the copy of the log file.
  - *View a log file in the Content Browser*. Available only for VWorks Plus, an administrator can use the browser to view files in the Secure Storage repository.

Viewing a log in the VWorks window

You can view the Main Log, Pipette Log, and Time Constraints Log in the VWorks window. When you exit the VWorks software, the log display is cleared and log files for the session are saved with the date-time stamp of the VWorks startup time for the given session.

To view a log in the VWorks window:

In the **VWorks** window, select **View > <log name>.** The log information appears is the corresponding log tab.



Viewing the log file contents

Log files are text files that have the following file naming conventions, where *date_time* in the file name is the startup time of the given VWorks session.

Type of log	File name
Main Log	VWorks_log(*date_time*).log
Pipette Log	VWorks_pipette_log(*date_time*).log
Time Constraints Log	VWorks_time_constraints_log(*date_time*).log

To view a copy of a log file:

1. Save a copy of the log file to a location of your choice. See "Validating and saving copy of log file" on page 716.
2. To view the log file contents, locate and select the saved copy of the log file in the Windows Explorer window.

   Open the file using a text editor such as Notepad**.**

(VWorks Plus only) To view a log file in the Content Browser:

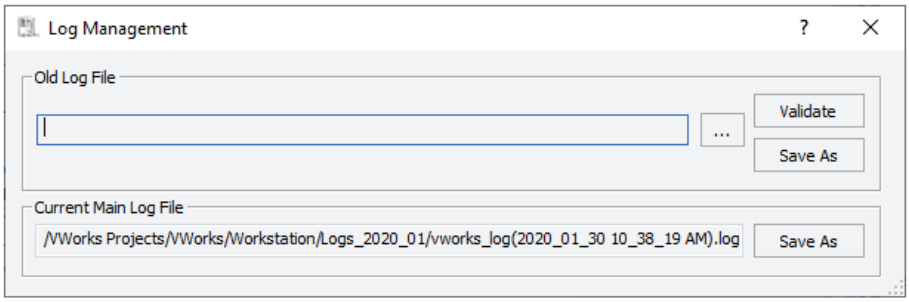
1. Start the Content Browser:

   Click **All apps > Agilent Technologies > Content Browser**.
2. Log in using your user name and password.
3. Locate the log file folder in the Repository, for example, /VWorks Projects/VWorks/Workstation/Logs.
4. Double-click the log file, <*log name*>log(*date_time*).log, to display the file details for the corresponding log.

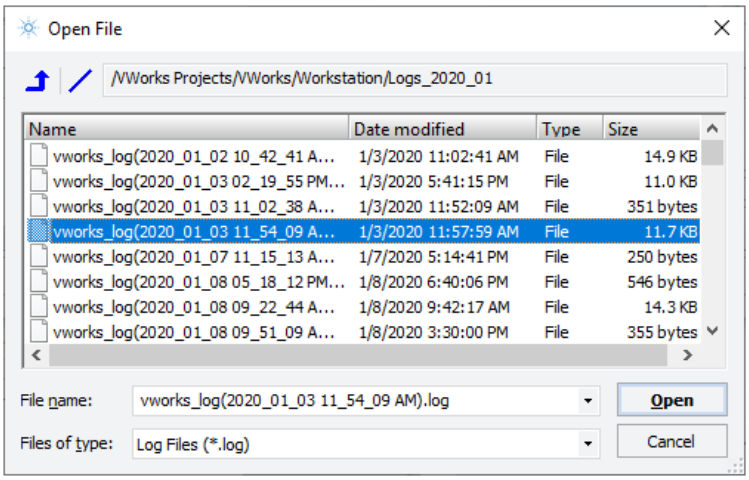
Validating and saving copy of log file

To validate a log file and save a copy to another location:

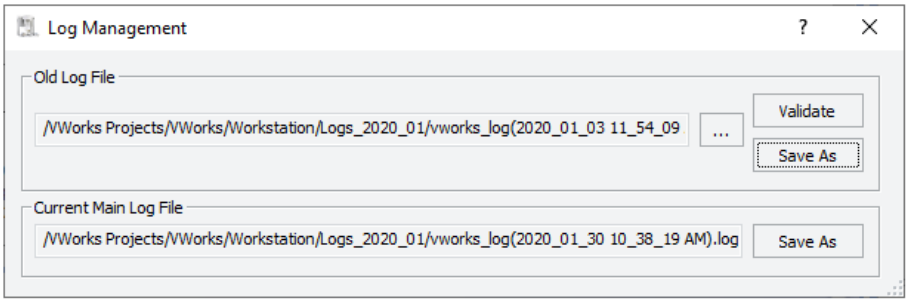
1. In the **VWorks** window, click **Tools > Log Management.** The Log Management dialog box opens.



2. Under **Old Log File**, click ... . The Open File dialog box opens.



3 Locate and select the log file. The path of the selected file appears in the Log Management dialog box.



Note: The log file path locations are listed in the Options dialog box.

4 In the **Old Log File** area, do one of the following:

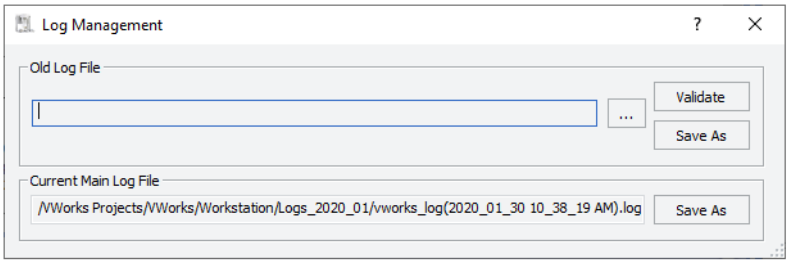
- Click **Save As**. The software validates the selected file to determine if it has been altered or corrupted.
  - If the validation is successful, the Save Old Log File As dialog box opens. Type a file name, select a location, and click **Save**.
  - If the validation is not successful, an error message appears and the save as procedure is aborted.
- Click **Validate**. The software validates the selected file to determine if it has been altered or corrupted.

  The software checks the log file and displays a message that explains whether the validation is successful.

Saving a copy of current Main Log

To save a copy of the Main Log for the active session:

1 In the **VWorks** window, click **Tools > Log Management.** The Log Management dialog box opens.



2 Under **Current Main Log File**, verify that the log path appears, and then click **Save As**.

3 In the **Save Current Main Log File As** dialog box, type a file name, select a location, and click **Save**.

Main Log contents

The Main Log records all available event and error information. The Main Log contains the following information:

Column name	Description
Timestamp	Time and date of the event or error.
Class	The type of event or error message: • *Info*. General information. For example, System start up, or Simulation mode toggled on. • *Event*. A software action. For example, Scheduler started, or Move plate. • *Error*. An error that can stop the software and must be resolved. For example, Location incompatible with labware. • *Warning*. An error that might permit the software to continue. For example, Task requires that tips be on the pipette head.
Device	Device at which the event or error occurred.
Location	Location where the event or error occurred. For example, Bravo location 9.
Process	Name of the protocol process and subprocess that is running.
Task	Task at which the event or error occurred.
Description	Description of the action that is being recorded.
File name	Name and location of the protocol file.
Session ID	Login session number.

Pipette Log contents

The Pipette Log records all pipetting events and error information. The Pipette Log contains the following information:

Column name	Description
Timestamp	Time and date of the pipetting event or error.
Class	The type of event or error message: • *Event*. A software action. For example, when a protocol is started, the event is logged. • *Transfer*. A pipetting event.
Session ID	Login session number.
Volume	The volume of liquid transferred.
Aspirate Location	The location at which the Aspirate task occurred.
Aspirate Selection	The wells from which fluid was drawn for the Aspirate task.
Dispense Location	The location at which the Dispense task occurred.
Dispense Selection	The wells into which fluid was dispensed.
Description	The description of the action that is being recorded.
File name	The name and location of the protocol file.

Time Constraints Log contents

The Time Constraints Log records any time-limited task events and errors. If you did not specify time constraints, no information appears in the log.

The Time Constraints Log contains the following information:

Column name	Description
Timestamp	Time and date of the event or error.
Class	The type of event or error message: • *Info*. General information. For example, System start up, or Simulation mode toggled on. • *Event*. A software action. For example, Scheduler started, or Move plate. • *Error*. An error that can stop the software and must be resolved. For example, Location incompatible with labware. • *Warning*. An error that might permit the software to continue. For example, Task requires that tips be on the pipette head.
Session ID	The login session number.

Column name	Description
Start Process	The name of the protocol process or subprocess that contains the first of the two dependent tasks. This field also includes the instance number to help you identify the plate instance. For example, Process - 1 1 means Process - 1, instance 1. Start Process is paired with Start Task to identify the first of the two dependent tasks.
Start Task	The task number that identifies the first of the two dependent tasks. For example, 1 is the first task in the process, and 5 is the fifth task in the process. Start Task is paired with Start Process to identify the first of the two dependent tasks in the protocol.
End Process	The name of the protocol process or subprocess that contains the second of the two dependent tasks. This field also includes the instance number to help you identify the plate instance. For example, Process - 3 1 means Process - 3, instance 1. End Process is paired with End Task to identify the second of the two dependent tasks.
End Task	The task number that identifies the second of the two dependent tasks. For example, 3 is the third task in the process, and 10 is the tenth task in the process. End Task is paired with End Process to identify the second of the two dependent tasks in the protocol.
Target interval	The time specified in the Edit Time Constraints dialog box.
Allowed deviation	The time tolerance specified in the Edit Time Constraints dialog box.
Actual interval	The length of time of the first task during the run.
Actual deviation	The difference between the Target interval and the Actual interval.
Error	The result of the following: |Actual deviation| – |Allowed deviation| If the result is less than 0, Error is set to 0. For example: • Actual deviation is 66. Allowed deviation is 60. So Error is 66 – 60, or 6. • Actual deviation is 30. Allowed deviation is 60. So Error is 30 – 60, or –30. Because the result is less than 0, Error is set to 0.
Description	The description of the action that is being recorded.
File name	The name and location of the protocol file.

Log text colors

Text in the logs appear in the following colors:

Color	Description
Grey	Standard events with a date stamp or operator-added notes
Blue	Liquid transfer events
Orange	Warnings
Red	Error
Green	General information
Purple	Debug information (shown only if the Debug log level is greater than 0 in Tools > Options)

Related information

For information about...	See...
Audit trails and records of interest	"Audit trails and audit trail reports" on page 15 *VWorks Automation Control Setup Guide*
Setting system-wide options	"Setting general and view options" on page 301
Exporting and importing protocol and associated files	"Exporting and importing protocols and associated records" on page 703
Contacting Agilent and reporting problems	"Reporting problems" on page 743

Resolving device initialization errors

Device initialization process

When you start the VWorks software, the software loads the driver files (plugins) for all the devices in the automation system. A record of this process is displayed in the Main Log.

When you open a protocol, the device file associated with that protocol opens. The device file tells the software which devices are connected to the system. For some devices, an initialization step tests the communication between the VWorks software and the device.

Resolving initialization errors

During the device initialization process, the software displays the list of devices it is expecting to find. Devices are removed from the list as the software determines that the devices are ready.

If a problem occurs during initialization, an error message appears and explains the problem.

To resolve the problem:

1. Make sure the device is turned on.
2. Make sure the communication cable is connected properly.
3. Make sure the communication cable is connected to the correct COM port.
4. Check the device profile to make sure it is set up correctly for communication.
5. If applicable, follow the instructions in the error message to fix the communication problem.
6. Click **Retry** to re-initialize the device.
7. If the problem persists, contact Agilent Technical Support.

Related information

For information about...	See...
Turning on the device	Device user guide
Editing the device profile	Device user guide
Contacting Agilent and reporting problems	"Reporting problems" on page 743

Resolving compilation error messages

Compilation warnings and error messages

During the protocol-compiling process, the software reports errors in the Main Log. You can use the information to troubleshoot the protocol.

The software displays two types of messages during the compiling process:

- Warning messages
- Error messages

Note: If you are logged in with technician, operator, or guest privileges, you are unable to continue with the protocol. If you are logged in with administrator privileges, the dialog box allows you to run the protocol despite the errors.

Warning messages

A warning message alerts you to an error that should be fixed. If the error is left unresolved, the software can still continue. For example, a pipetting task requires that tips be on the pipette head, but there was no Tips On task preceding the pipetting task.

Error messages

Error messages alert you to situations where a protocol or device will fail. You must resolve the problem to continue the run.

Errors are generated when:

- Operating parameters are out of range, denoted by red text in the Task Parameters area.
- A task wants to use a labware that does not exist in the system.
- Volumes in pipette steps do not match, such as when a dispense volume is greater than a previous aspirate volume.
- A Signal task has no associated Wait For task.

Related information

For information about...	See...
Compiling protocols	"Compiling the protocol" on page 77
Viewing logs	"Viewing logs" on page 714
Contacting and reporting problems	"Reporting problems" on page 743

Disabling and enabling tasks

When to disable tasks

You can disable tasks when you test run a protocol. Disabling one or more tasks allows you to:

- Run a protocol but skip tasks that use problem devices or locations. Doing so allows you to verify that the rest of the protocol is working properly while you troubleshoot the problem devices and locations.
- Skip tasks that are not needed for certain runs. You can reuse the same protocol for a number of situations without creating a number of different protocols.

Note: You can also skip tasks by disabling the device assigned to the task in the device file. This allows you to run alternative protocol configurations using a single device file. See "Disabling and enabling a device in the device file" on page 293 for more information.

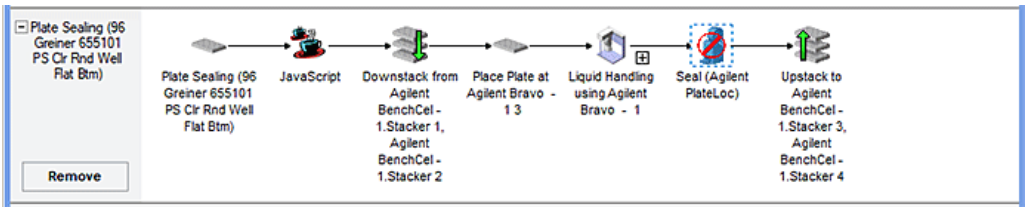
Disabling tasks

To disable a task:

1. Right-click the task that you want to disable.
2. In the shortcut menu that appears, select **Disable task.**

   A red circle and strike line appears on the icon to indicate that it is a disabled task. The software will bypass the task when the protocol is running.

   In the following example, the Seal task is disabled in the protocol.



Enabling tasks

You can enable tasks individually or you can enable all tasks using one command.

To enable a task:

1. Right-click the disabled task.
2. In the shortcut menu that appears, select **Enable task.**

To enable all tasks in a protocol:

1. Right-click any task in the protocol.
2. In the shortcut menu that appears, select **Enable all tasks.**

Related information

For information about...	See...
The Halt on barcode misreads option in the Options dialog box	"Setting general and view options" on page 301
Viewing logs	"Viewing logs" on page 714
Running protocols in simulation mode	"Simulating the protocol run" on page 80

Using breakpoints to monitor and troubleshoot tasks

About breakpoints

A breakpoint is a point in the protocol where you want the system to pause operation. Adding a breakpoint at a task pauses the run or simulation and opens the Debugger dialog box before the system performs the actual task. The Debugger dialog box provides JavaScript information for the corresponding task.

Breakpoints enable you to do the following:

- You can monitor parameter values and verify the actions of certain tasks during simulation or a dry run, and change the values if necessary.

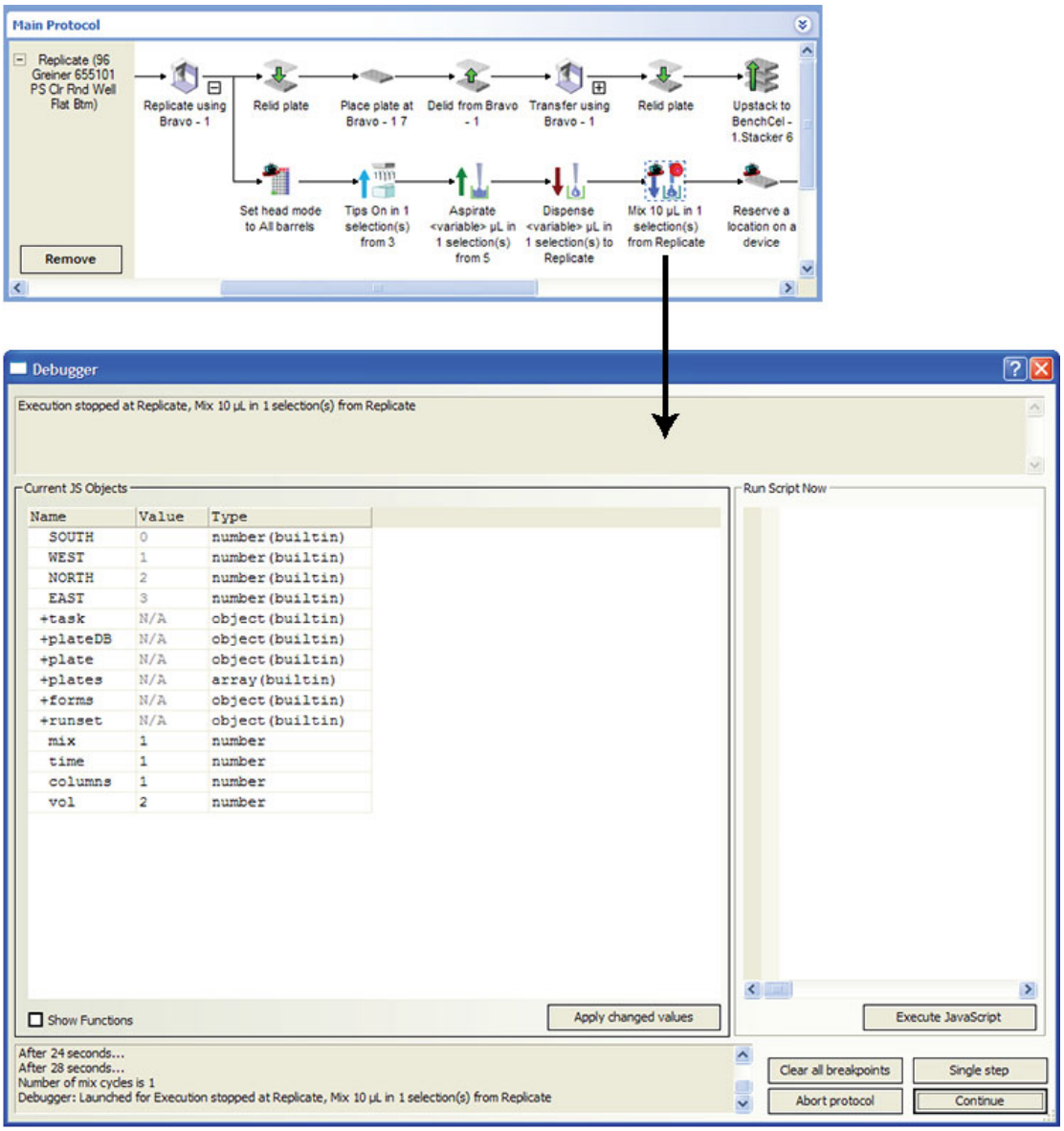
  For example, you can add a breakpoint at an aspirate task to pause the system just before the aspirate task starts. During the pause, you can verify the task parameter values in the JavaScript engine, after the task's script executes, and before the parameters are applied to the task.
- If you declare your own JavaScript variables for certain tasks, you can monitor the values by placing breakpoints at those tasks.
- You can troubleshoot tasks and fine-tune task parameter values.
- The Debugger dialog box also enables you to write and execute scripts in real time that are not necessarily associated with the selected task.

During the protocol run, the following sequence occurs when the software reaches a breakpoint:

Step	Description
1	The protocol pauses at the task with the breakpoint.
2	The JavaScript associated with the selected task executes.
3	The Debugger dialog box opens and displays the JavaScript for that task.
4	In the Debugger dialog box, you can modify values in the JavaScript engine.
5	The software applies the relevant task parameters from the JavaScript engine to the scheduler. If you changed task parameter values in the Debugger dialog box, the task will use the updated values when you continue the run.
6	You can continue running the selected task and the rest of the protocol or abort the run.

Note: If the breakpoint is on a task within a loop or if the protocol is run multiple times in one session, the protocol will pause each time it reaches that task.

Figure A breakpoint in a protocol and the corresponding Debugger dialog box



Adding breakpoints

You can add as many breakpoints as you want in a protocol. In addition, you can add breakpoints while a protocol or simulation is running.

To add a breakpoint:

1. In the **Protocol** area, right-click the task at which you want the breakpoint.
2. In the shortcut menu that appears, click **Set breakpoint.** A red dot appears on the task icon.

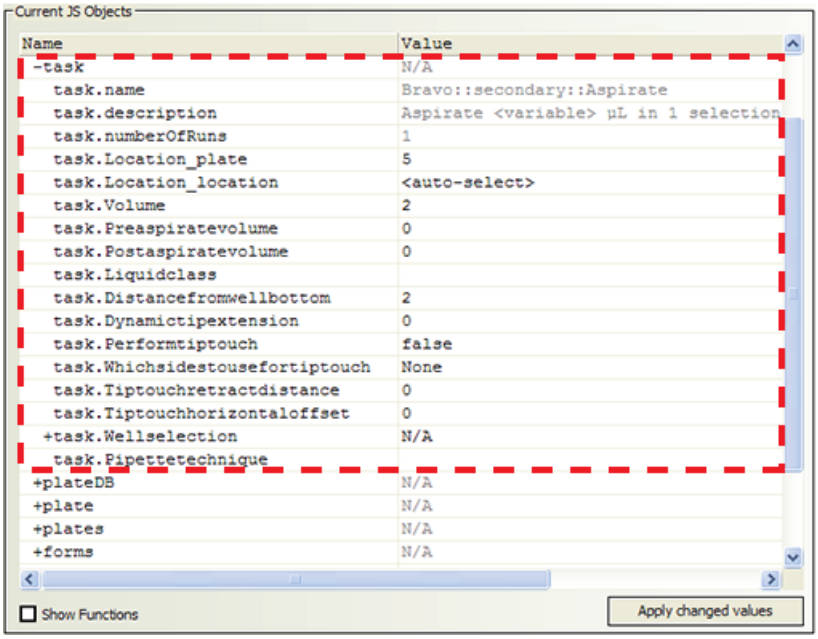


Using a breakpoint during a simulation or a dry run

To use a breakpoint in a protocol:

1. Start a simulation or dry run of the protocol that contains the breakpoint. When the software reaches the task that has the breakpoint, the Debugger dialog box opens.
2. Under **Current JS Objects** in the Debugger dialog box, look in the **Name** column for the object that you want to view. Click the **+** symbol next to the object name to view the corresponding properties.

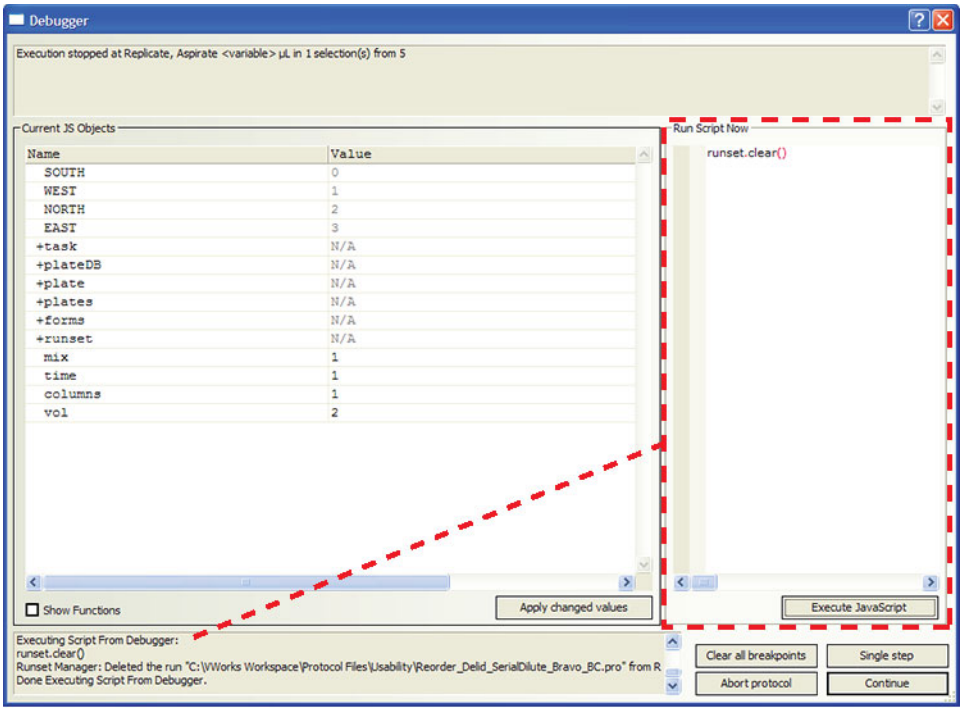
   For example, the following figure shows the task object properties for an aspirate task at which a breakpoint was set.



   Note: To show the functions (methods) as well as the properties associated with the JavaScript object, select **Show Functions**, and then expand the object in the **Name** column.
3. In the **Value** column, change a property value, if required. For example, you might change the `task.Volume` value from 2 to 3.
4. To apply any changed values without closing the Debugger dialog box, click **Apply changed values**.
5. If you want to run a script from the Debugger dialog box:
   a. Type the script in the **Run Script Now** area.
   b. Click **Execute JavaScript**.

   The status of the script appears in the bottom left of the Debugger dialog box.

   For example, you could type the script `runset.clear()to` clear all the entries in the runset manager, except for currently running protocol.



6 Click the button in the Debugger dialog box that corresponds to the command you want to use next:

Button	Description
Clear all breakpoints	Removes all breakpoints in the protocol and continues the run.
Single step	Permits the protocol to perform one task at a time, pausing before each task.
Continue	Resumes the protocol run until the next breakpoint is reached.
Abort	Aborts the protocol run.

Removing breakpoints from an open protocol

You can remove breakpoints individually or you can remove all breakpoints with one command.

To remove a specific breakpoint:

1 Right-click the task that has a breakpoint.

2 In the menu that appears, click **Remove breakpoint.**

To remove all breakpoints in a protocol:

1 Right-click any task in the protocol.

2 In the menu that appears, click **Clear all breakpoints.**

Related information

For information about...	See...
Compiling protocols	"Compiling the protocol" on page 77
Simulating a protocol run	"Simulating the protocol run" on page 80
JavaScript objects in the VWorks software	"Creating a protocol: advanced topics" on page 89
Tracking the run progress of instances, processes, or devices	"Tracking the run progress of instances or devices" on page 330
Contacting Agilent Technologies and reporting problems	"Reporting problems" on page 743

Resolving barcode reader error messages

About this topic

The information in this topic applies only to the following barcode readers:

- Robot barcode readers
- Barcode readers that are installed on a device such as a platepad

The information in this topic does not apply to the Microplate Labeler.

Causes of barcode reader errors

Errors generated by robot barcode readers and barcode readers that are installed on devices are generally caused by:

- Poor printing or label placement such that the barcodes are not perpendicular to the barcode laser read line
- Poor label placement such that the barcode cannot be fully read
- Missing or damaged barcode labels
- Wrong barcode labels

Note: The system does not attempt to read a barcode unless you specified that the labware has a barcode label. See "Setting plate parameters" on page 59 for instructions.

Recovering from barcode reader errors

Your ability to resolve the barcode reader errors depends on whether the **Halt on bar code misreads** option is selected in the Options (Tools > Options) dialog box.

If the **Halt on bar code misreads** option is selected:

- The protocol pauses.
- The error message is recorded in the Main Log.

- The error message dialog box opens and allows you to type the correct barcode. With the correct barcode, the software continues the run.

If the **Halt on bar code misreads** option is not selected:

- The error message is recorded in the Main Log.
- The protocol continues without pausing, so you cannot correct from the error.

A quarantine response is set up in the Error Library for a default set of barcode reader error messages. The quarantine response allows the system to continue running the protocol even though it is unable to resolve problems with the labware. For a description of the quarantine response, see "Setting up automated responses" on page 736.

Related information

For information about...	See...
The Halt on bar code misreads option in the Options dialog box	"Setting general and view options" on page 301
Viewing logs	"Viewing logs" on page 714
Contacting Agilent and reporting problems	"Reporting problems" on page 743

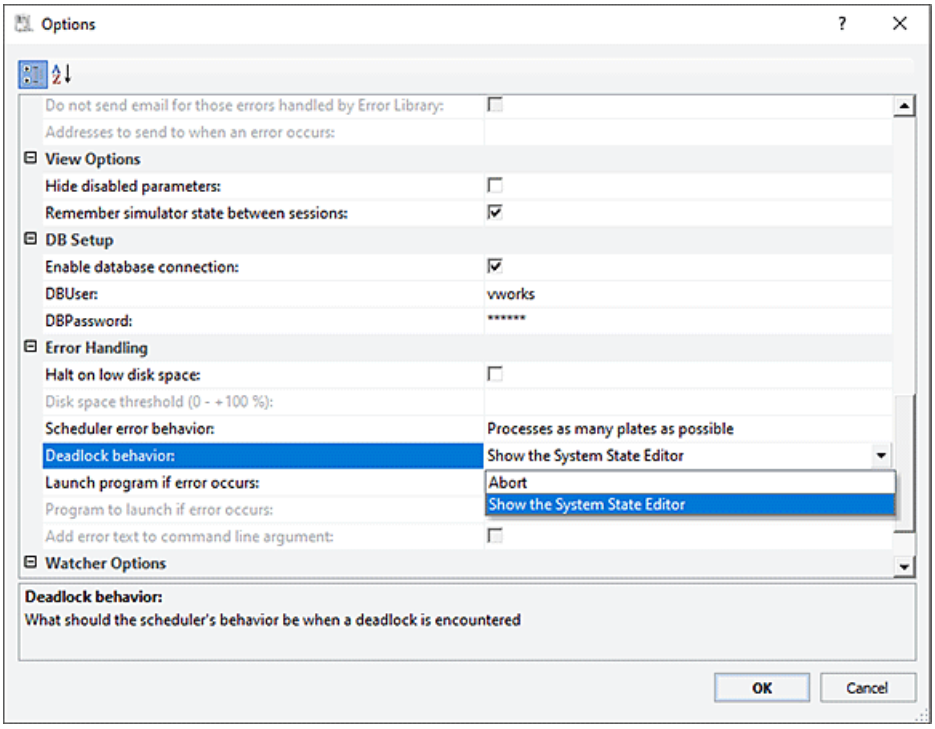
Recovering from deadlocks

About deadlocks

A deadlock is an error that occurs when the number of locations available in the system is less than the number of microplates in the system. Because the microplates cannot move to the expected locations, the protocol either aborts or pauses, depending on the **Deadlock behavior** selection in the Tools > Options dialog box.

About the System State Editor

In the Tools > Options dialog box, if the **Deadlock behavior** is **Show the System State Editor,** the run will pause when a deadlock error occurs.



During the pause the System State Editor dialog box appears.

In the System State Editor dialog box, you can:

1. View the status of the devices and locations to assess and correct the causes of the errors. For example, if the device status indicates that a labware at a particular location is causing device problems, you can physically remove that labware from the system and restore the device to its operational state.
2. Specify in the software that a labware has been removed from certain locations or from the system. For example, after you physically remove the labware that is causing problems, you need to indicate in the software that the labware is removed and its location is now ready to accept new labware.
3. Edit or reset the status of devices and locations. If you fixed a device problem, you can indicate that the device is now operational and ready to accept labware.
4. Continue the protocol run.

If you determine that the causes of the deadlock error cannot be fixed, you can choose to abort the run or allow the existing labware to finish processing but not introduce new labware. For more information about these options, see "Pausing the run" on page 337.

You can also open the System State Editor when you pause all runs. For instructions, see "Pausing the run" on page 337.

Viewing the device and location status

To view the device and location status:

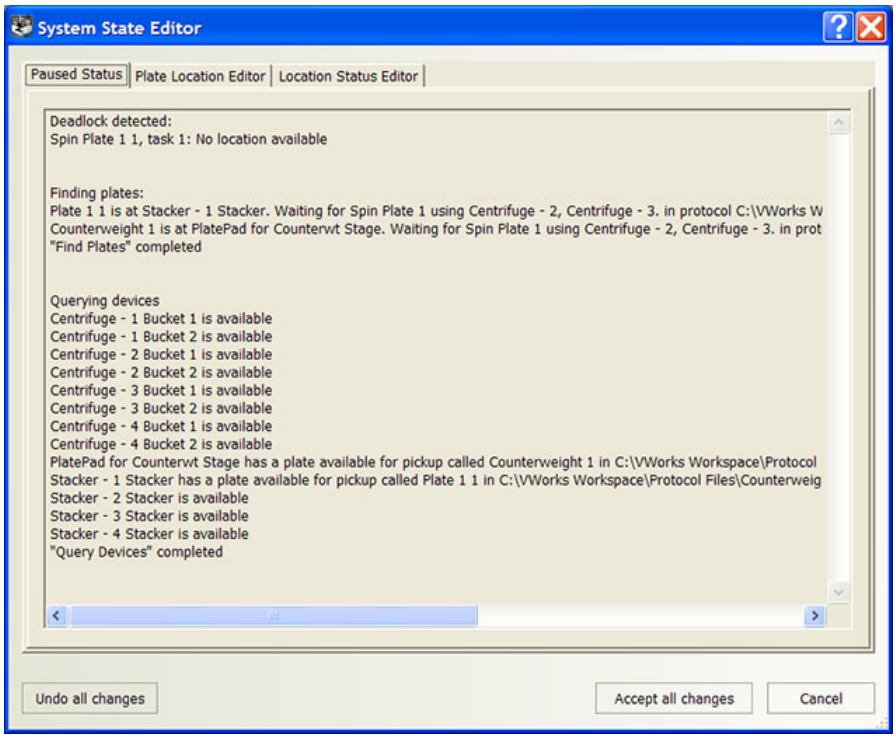
1. In the **System State Editor** dialog box, click the **Paused Status** tab.
2. Review the information in the **Paused Status** tab.

   The Paused Status tab contains the following information:

   - Causes of the deadlock error

- Current locations of the labware in the system and the expected locations when the deadlock error occurred
- The current status of all the devices and locations in the system

Note: The information in the Paused Status tab can be appended to the Main Log file. To do this, select **Tools > Get Status.**



Moving or removing labware from locations

The Plate Location Editor tab lists:

- All active process plates in all running protocols
- The current locations of the process plates

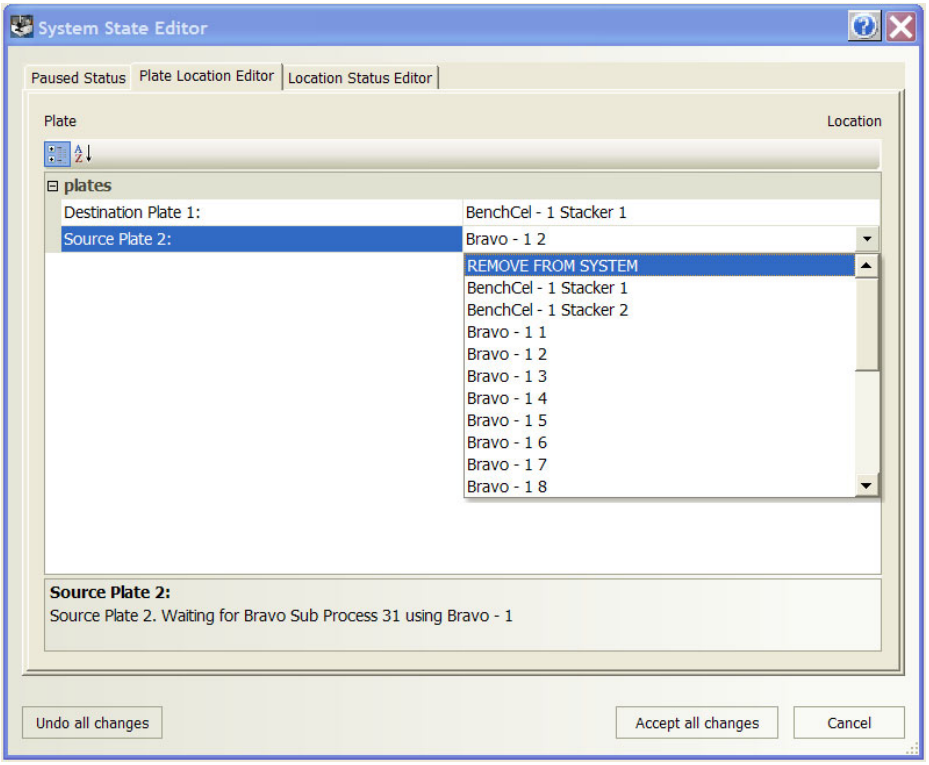
IMPORTANT

The list does not include finished process plates. To see the finished process plates, click the Location Status Editor tab.

To indicate that labware is moved or removed from locations:

1. In the **System State Editor** dialog box, click the **Plate Location Editor** tab.
2. To change the location of a process plate, select the new location from the list. If you want to indicate that a process plate was removed from the system, select **REMOVE FROM SYSTEM**.

   Note: The software prevents you from selecting one location for two different process plates. If you select a location that is already occupied by another process plate, the software will exchange the two locations.



Editing the status of the devices and locations

CAUTION **Do not change the status of device locations without fully knowing the state of all process plates and device locations. An arbitrary change can cause crashes when you resume the run. If you are uncertain of the state of the process plates and device locations, contact Agilent Technical Support.**

The Location Status Editor tab displays all labware in running protocols in:

- The list of every device and associated locations
- The status of the locations

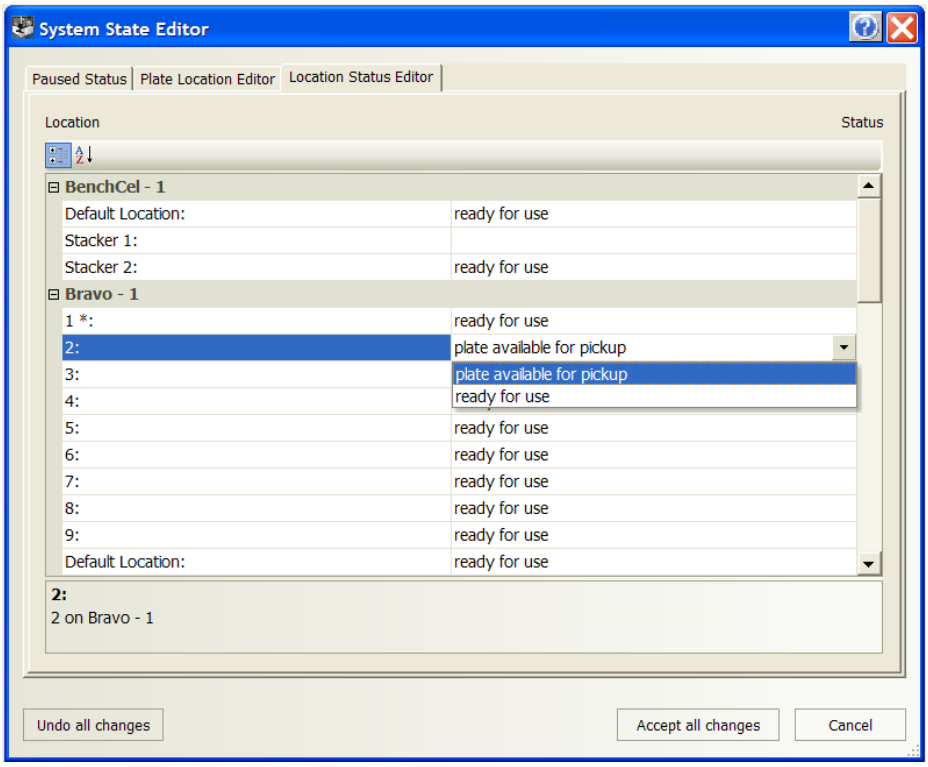
Finished process plates appear in this tab. You can use the selections in this tab to remove finished process plates.

CAUTION **Make sure finished process plates will not be used by downstream processes before removing them.**

IMPORTANT Except to remove finished process plates, you do not need to change the status of device locations. The changes you made in the Plate Location Editor tab are automatically reflected in the Location Status Editor tab. For example, if you removed a process plate from a location in the Plate Location Editor tab, the status of that location will change from Plate available for pickup (location is occupied) to Ready for use (location is available for use)

To remove a finished process plate:

1. In the **System State Editor** dialog box, click the **Location Status Editor** tab.
2. Find the finished process plate location, and then select the new status from the list.



Status	Description
Plate available for pickup	Retains the finished process plate in the system.
Ready for use	Removes the finished process plate from the system.

Continuing the protocol run

After you specify the labware movement or removal and the new device and location status:

1. In the **System State Editor** dialog box, click **Accept all changes.**
2. In the **Scheduler Paused** dialog box, click **Continue** to resume the run.

   The Scheduler Paused dialog box also provides other options. For details, see "Pausing the run" on page 337.

Related information

For information about...	See...
Pausing a run	"Pausing the run" on page 337

For information about...	See...
Monitoring a run	"Tracking the run progress of instances or devices" on page 330
Viewing logs	"Viewing logs" on page 714
Tracking the run progress of instances, processes, or devices	"Tracking the run progress of instances or devices" on page 330
Reporting problems	"Reporting problems" on page 743

Setting up automated error responses

About the Error Library

If you are a VWorks administrator or technician, you can use an error library to:

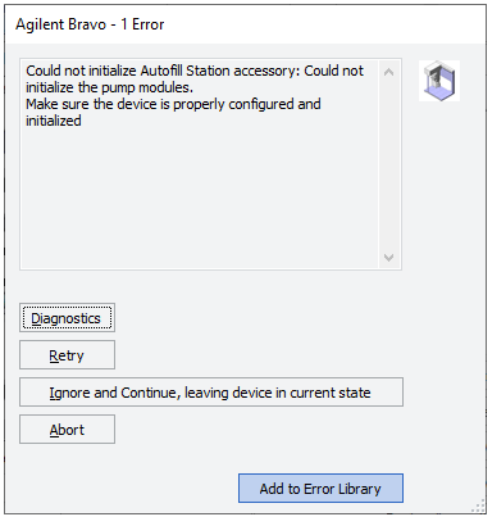
- Display or hide certain error messages during a protocol run. Hiding minor errors reduces the number of interruptions in the protocol run.
- Automate error recovery responses for selected errors to reduce the number of manual interventions in a protocol run.
- Track the frequency of certain errors for troubleshooting, establishing standard automated recovery responses, or quality-control reporting.

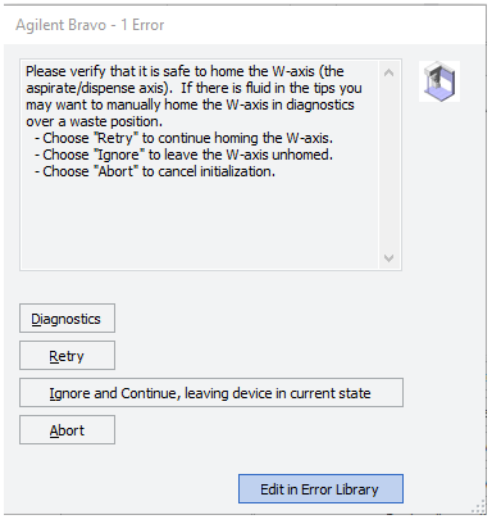
For example, you can set up a quarantine response for barcode reading, incorrect labware orientation, and wrong labware type errors. The quarantine response allows the system to continue running the protocol even though it is unable to resolve problems with the labware.

Adding errors to the library

You can add error messages to the error library to set up display options and automated responses. When you initialize a device or run a protocol and an error dialog box opens, click **Add to Error Library** to add the displayed error message to the error library. See the following figure.

If the error message is already in the error library, the Edit in Error Library button is available in the error dialog box.





To set up quarantine responses, you can perform dry runs to generate the desired error messages for the error library. For example, you can use labware that have poorly printed or missing barcode labels, load labware backwards in stackers, or load the wrong labware type in stackers.

Note: The software automatically records the frequency of the error in the error library.

Setting up automated responses

Before you start, ensure the following in the Options dialog box:

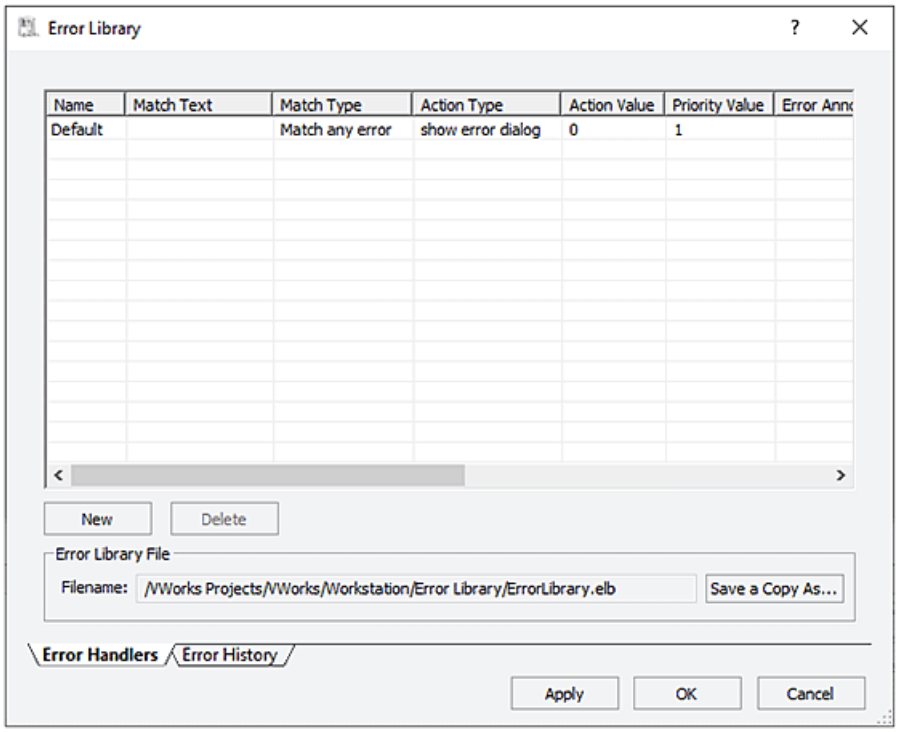
- The correct **Error library path** is specified. For details, see "Setting file paths for libraries" on page 299.
- The **Enable error library** check box is selected.

Note: To open the Options dialog box, click Tools > Options.

To set up error responses:

1 In the **VWorks** window, click **Tools > Error Library.** The Error Library dialog box opens.

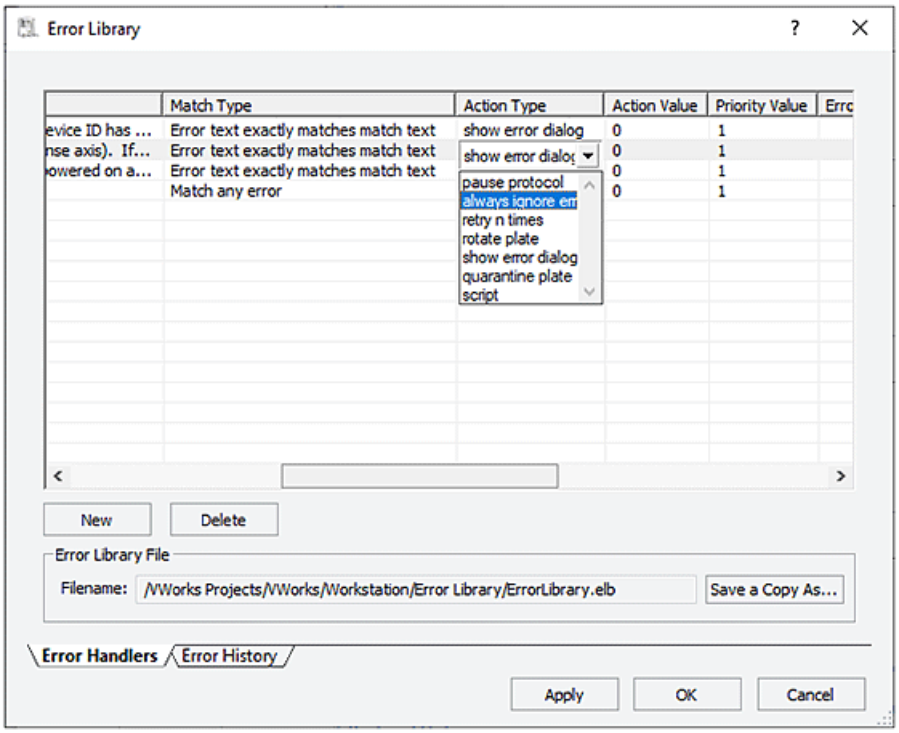
Note: The software opens the file for the error library specified in the Options dialog box.



2 In the **Error Handlers** tab, set the following parameters to identify the error handlers. An error handler is a set of conditions that define a specific response to an error.

Parameter	Description
Name	The name of the error handler. Type a name that can help you to identify it quickly in the log. *Note:* The Default error handler is used to define the default error response. It can be edited, but it cannot be deleted from the error library.
Match Text	The text of the error message that must be matched to activate the error handler. You can modify the text to set it up for match-text filtering.
Match Type	The match-text filtering to use: • Error text exactly matches Match Text • Error text contains Match Text

3 Set the following to determine the error response:



Parameter	Description
Action Type	The automated error response to use whenever the text-matching conditions are met: • *Pause protocol*. Pauses the protocol. • *Always ignore error*. Ignores the current command or task and continues to the next command or task in the protocol sequence. • *Retry n times*. Attempts to restart the current command or task in the run. The number of attempts is specified by the Action Value. • *Rotate plate*. Places the labware at a device that can rotate it to correct its orientation. • *Show error dialog*. Displays the error dialog box to allow the operator to determine the error response. • *Quarantine plate*. Moves the labware that caused the error to a quarantine location and continues the run. • *Script*. Runs the script specified in the **Script to Execute** box. See step 6.
Action Value	The number of times to restart the current command or task.

4 Type the **Priority** value in case the text from two errors matches the conditions in two handlers. The larger the value, the lower the priority.

The software sets the overall priority as follows:

a Exact error text match

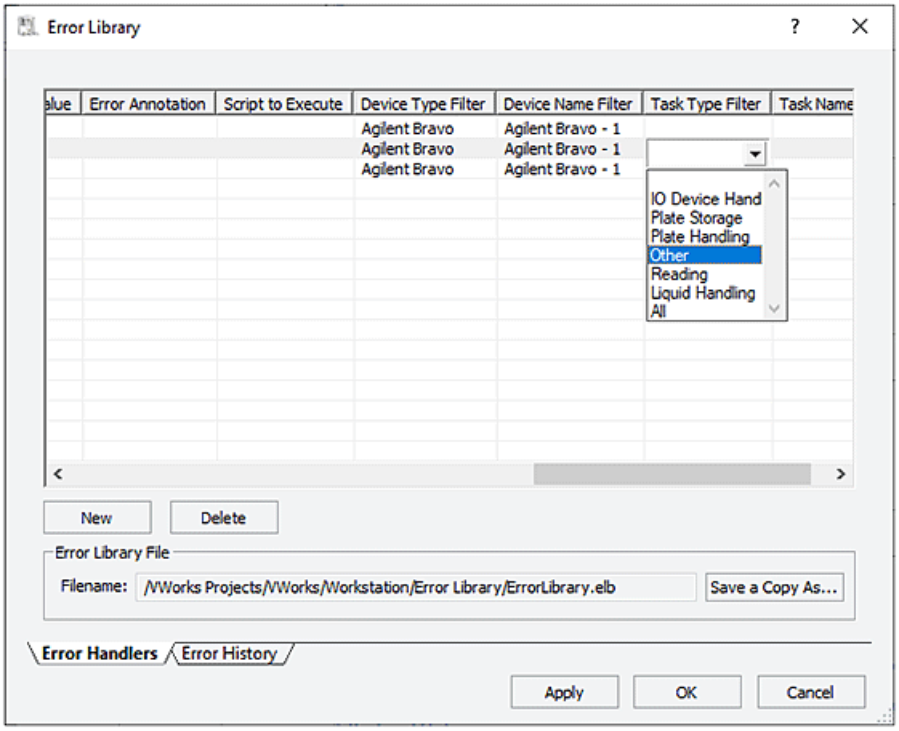
b Partial error text match (longer text matches are preferred over shorter text matches)

c Larger Priority value (for example, 1 has higher priority than 2)

5 Type additional notes about the error in the **Error Annotation** box. The text in this box is displayed with the original error text. You can use this field to explain how to fix the error.

6 Type the script you want to run in the **Script to Execute** box. This box is only active if you have selected **Script** for the **Action Type** in step 3.

7 Set up additional filters for the error message. You can select the desired value for each of the parameters in the following table. If you do not want to set up a filter for one or more of the parameters, select the empty value from the parameter list.



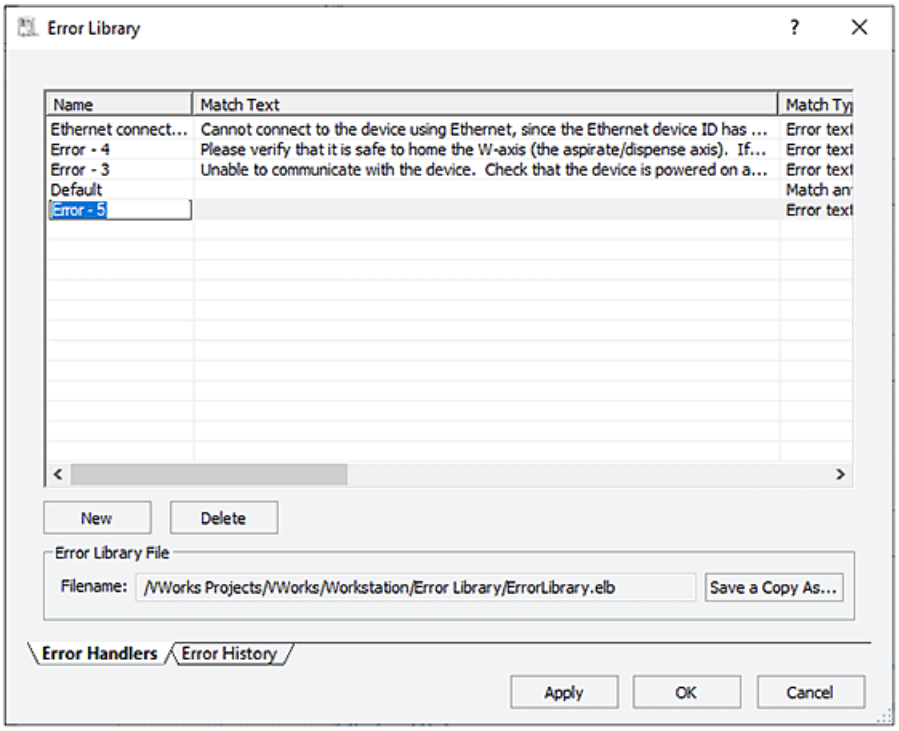
Parameter	Description
Device Type Filter	The device type, such as Agilent Bravo.
Device Name Filter	The device name that distinguishes two devices of the same type, for example, Agilent Bravo-1 and Agilent Bravo-2.
Task Type Filter	The task category, for example, Liquid Handling tasks.
Task Name Filter	The name of the task, for example, Aspirate.

8 To save the changes click **Apply**. Or, click **OK** to save the changes and close the Error Library dialog box.

Creating a new error handler

To create a new error handler:

1 In the **Error Library** dialog box, click **New.** A new row appears in the error handler table with a default name, Error - *n,* where *n* represents the numeric sequence in which it was created.



2 Follow the instructions in "Setting up automated responses" on page 736 to set up the new error handler.

Note: In a newly created handler, the Match Text field is empty. Type the error text that you want to use to set up the handler.

3 To save the changes click **Apply**. Or, click **OK** to save the changes and close the Error Library dialog box.

Deleting an error handler

To delete an error handler:

1 In the **Error Library** dialog box, select the error handler you want to delete.

2 Click **Delete.** The error handler is removed from the table.

3 To save the changes click **Apply**. Or, click **OK** to save the changes and close the Error Library dialog box.

Saving a copy of the error library

To save a copy of the error library file:

1 In the **Error Library File** area, click **Save a Copy As.** The Save File As dialog box opens.

2 Type the new file name, and then click **Save**.

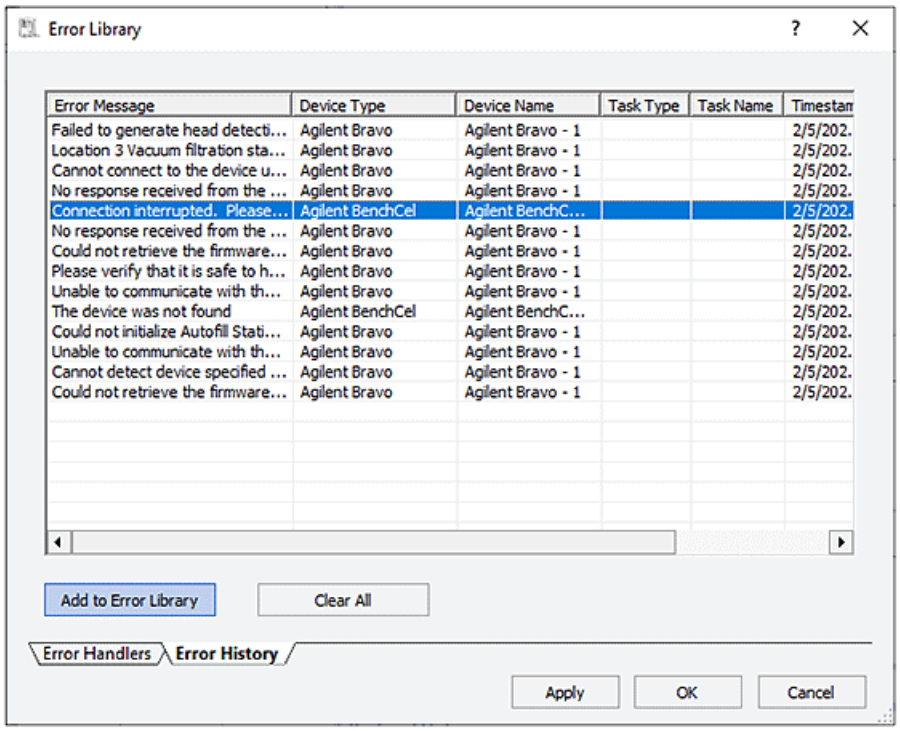
Note: By default, error library files are saved in the /VWorks Projects/VWorks/Workstation/Error Library folder.

Viewing and tracking error occurrences

You can view all the errors that have occurred during the protocol run that have not yet been assigned an error handler. The error library records the frequency of their occurrences and other information to help you determine whether you want to add them to the library and what automated response to specify.

To view errors that are not yet set up with error handlers:

1 In the **Error Library** dialog box, click the **Error History** tab. The list of error messages appears in a table.



2 To sort the list of errors in the table by a given parameter, click the corresponding column heading in the tab:

Column heading	Description
Error Message	The text of the error message.
Device Type	The type of device on which the error occurred, for example, Agilent Bravo.
Device Name	The name of the device on which the error occurred. The name distinguishes two devices of the same type, for example, Agilent Bravo-1 and Agilent Bravo-2.
Task Type	The category of the task in which the error occurred, for example, Liquid Handling tasks.

Column heading	Description
Task Name	The task in which the error occurred. For example, Aspirate.
Timestamp	The date and time at which the error occurred.
Error Times	The number of times the error occurred within the protocol.
File Name	The name of the device file containing the device.

To add an error message to the error library file:

1. Select the error message in the table.
2. Click **Add to Error Library.**
3. Follow the instructions in "Setting up automated responses" on page 736.

To clear the history table:

Click **Clear All.**

Related information

For information about...	See...
Viewing logs	"Viewing logs" on page 714
Tracking the run progress of instances, processes, or devices	"Tracking the run progress of instances or devices" on page 330
Contacting Agilent and reporting problems	"Reporting problems" on page 743

Reporting problems

Contacting Technical Support

If you find a problem with the VWorks software, contact Agilent Technical Support. For contact information, go to https://www.agilent.com/en/contact-us/page

Reporting hardware problems

When contacting Agilent, make sure you have the serial number of the device ready.

Reporting software problems

When you contact Agilent Technical Support, make sure you provide the following:

- Short description of the problem
- Relevant software version number (for example, automation control software, diagnostics software, and firmware)
- Error message text (or screen capture of the error message dialog box)
- Relevant files, such as log files

Reporting user guide problems

If you find a problem with this user guide or have suggestions for improvement, send your comments in an email to documentation.automation@agilent.com.

Related information

For information about...	See...
Exporting a protocol and associated files	"Exporting and importing protocols and associated records" on page 703
VWorks software components: protocol, device file, and so on	"Relationship of components in VWorks software" on page 5
Viewing logs	"Viewing logs" on page 714

A Quick reference

This appendix contains the following topics:

- "Menu commands" on page 746
- "Toolbar buttons" on page 752

If the VWorks menubar or toolbars are hidden, right-click the window, and select the view options from the shortcut menu.

Menu commands

File menu

Command	Description
New > Protocol	Creates a new protocol. See "Workflow for creating a basic protocol" on page 22.
New > Device	Creates a new device file. See "Creating a device file" on page 27.
New > Runset	Clears the Runset Manager tab so that you can create a new runset. See "Managing runsets" on page 317.
New > Form	Opens the Form Designer so that you can create a new form to run a protocol or runset. See "Workflow for creating or editing a form" on page 181.
Open	Allows you to locate and open files, such as protocol, device, runset, form, and Gantt files.
Close	Closes the displayed protocol, device, runset, or form file.
Save	Saves the changes in the protocol, device, or form file.
Save As	Opens the Save File As dialog box and allows you to create a copy of the existing protocol or device file and specify a new name.
Save All	Saves all protocol, device, and form files currently open in the software.
Save Runset	Opens the Save File As dialog box, which enables you to specify a file name and save the runset file.
Save Runset As	Opens the Save File As dialog box, which enables you to save the runset file using a different file name.
Recent .pro files	Opens a recently opened protocol file.
Recent .dev files	Opens a recently opened device file.
Recent .rst files	Opens a recently opened runset.
Recent .VWForm files	Opens a recently opened protocol form.
Print	Prints the selected protocol, device, or form file.
Print Setup	Opens the Print Setup dialog box and allows you specify printing options.
Print Preview	Allows you to preview the printout.

Command	Description
Import VZP File	Imports a .vzp file that contains a protocol, runset, or form file and associated components. See "Exporting and importing protocols and associated records" on page 703.
Export VZP File	Exports a protocol, runset, or form file and associated components and stores them in a .vzp file. See "Exporting and importing protocols and associated records" on page 703.
Import Misc File	Imports a file that was created in software other than the VWorks software, such as image files, JavaScript files, batch files, and so forth. Some of these miscellaneous files must be imported before they are available for use in the VWorks software. For details, see "Importing and exporting miscellaneous files" on page 49.
Exit	Quits the VWorks software.

Edit menu

Command	Description
Cut	Removes selected text and stores it in memory.
Copy	Copies selected text and stores it in memory.
Paste	Pastes the text that is currently stored in memory.

View menu

Command	Description
Menubar	Displays or hides the menubar. When it is hidden, press ALT+V, or right-click in the protocol or device file area and select Menubar to display it.
Standard Toolbar	Displays or hides the standard toolbar.
Control Toolbar	Displays or hides the control toolbar.
Workspace	Displays or hides the Workspace area.
Available Tasks	Displays or hides the Available Tasks area.
Available Macros	Displays or hides the Available Macros area. For details on macros, see "Using macros to create protocols" on page 265.
Main Log	Displays or hides the Main Log in the log and progress area.

Command	Description
Pipette Log	Displays or hides the Pipette Log in the log and progress area.
Time Constraints Log	Displays or hides the Time Constraint Log in the log and progress area.
Status Bar	Displays or hides the status bar below the log and progress area.
Progress	Displays or hides the Progress table in the log and progress area.
Runset Manager	Displays or hides the Runset Manager in the log and progress area. For details on managing runsets, see "Managing runsets" on page 317.
Full Screen Mode	Available only if a form is open in the VWorks window and the form contains a Toggle Full Screen control. The Full Screen Mode command changes the VWorks window to a full screen display of only the form.

Tools menu

Command	Description
Labware Editor	Opens the Labware Editor. For details, see the *VWorks Automation Control Setup Guide*.
Liquid Library Editor	Opens the Liquid Library Editor. For details, see the *VWorks Automation Control Setup Guide*.
Pipette Technique Editor	Opens the Pipette Technique Editor. For details, see "Specifying pipetting techniques" on page 691.
System State Editor	Available during a paused protocol run. Opens the System State Editor. Enables you to view the status of the devices and locations to assess and correct the causes of the errors. For details, see: • "Opening the System State Editor" on page 335 • "About the System State Editor" on page 730
Tip State Editor	Available if automatic tip selection is specified when using the Hit Pick Replication (Bravo) task. Opens the Tip State Editor.

Command	Description
Configure Labware	Allows you to assign labware to device locations.
Audit Trail Report (VWorks Plus only)	Allows you to generate audit trail reports for selected records of interest. For details, see the *VWorks Automation Control Setup Guide*.
Tamper Detection (VWorks Plus only)	Runs a check of records of interest that are stored in the Secure Storage repository to detect corrupted records and records that have been modified outside of the VWorks software. For details, see the *VWorks Automation Control Setup Guide*.
Backup Manager > Back up	Allows you back up system files, including the inventory databases, labware definitions and liquid classes, and pipette techniques. For details, see "Backing up and restoring VWorks files" on page 700.
Backup Manager > Restore	Allows you to restore a backed up copy of the system files, including the inventory databases, labware definitions and liquid classes, and pipette techniques. For details, see "Backing up and restoring VWorks files" on page 700.
Log Management	Allows you to validate a log file and save a copy to another location and to save a copy of the current Main Log while it is open. For details, see "Viewing logs" on page 714.
Hit Pick Format Wizard	Allows you to set up the fluid transfer from a source microplate to a destination microplate. For details, see "Hit Pick Replication (Agilent Bravo)" on page 526.
E-Signatures > Sign (VWorks Plus only)	Available only if a connection to the E-Signatures database is established. Opens the Pending Records/ Experiments For E-Signatures dialog box, which displays all the records and experiment IDs that are pending e-signatures. See the *VWorks Automation Control Setup Guide*.
E-Signatures > View (VWorks Plus only)	Available only if a connection to the E-Signatures database is established. Opens the VWorks E-Signature Viewer, which enables you to view the signatures for a selected record or experiment ID. See the *VWorks Automation Control Setup Guide*.
Open Hooks Plugin for...	There are no hooks plugin loaded.
Error Library	Opens the Error Library. For details, see "Setting up automated error responses" on page 735.
Macro Library	Opens the Save As Macro Library dialog box. For details, see "Using macros to create protocols" on page 265.

Command	Description
Gantt Charts	Available only when one or more protocols are running, or when one or more open protocols have been run in the current VWorks session. Allows you to visually monitor the real-time status of processes, plate instances, and devices. For details, see "Tracking the run progress of instances or devices" on page 330.
Get Status	Appends the current status to the Main Log. You can use this command at any time, including during a run.
Inventory Editor	Available only if a connection to the Inventory database is established. This command opens the Inventory Editor. See the *VWorks Automation Control Setup Guide*.
Experiments Editor	Available only if a connection to the Experiments database is established. This command opens the Experiments Editor. See the *VWorks Automation Control Setup Guide*.
Experiments Archive	Available only if a connection to the Experiments database is established. This command opens the Experiments Archive and enables you to locate and restore an archived experiment ID. See the *VWorks Automation Control Setup Guide*.
Change ROI state (VWorks Plus only)	Available only if no records of interest are open in the VWorks window. For users with administrator or technician privileges, This command allows you to change the development state (In Development, In Validation, and Released) of a record of interest. For details, see the *VWorks Automation Control Setup Guide*.
Manage IO	Not used.
Watcher is ON/OFF	Available only if the Watcher feature is configured. For details, see "Setting up and using the Watcher tool" on page 755.
Edit Form	Available only if a form is open in the VWorks window. Displays the Form Editor window that enables you to edit the form. For details, see "Workflow for creating or editing a form" on page 181.
Migration	Opens the Migration Wizard, which allows you to migrate files and registry information from VWorks 12.3, 13.0, 13.1.x into Shared Services storage for use in VWorks 14.0 or later. See the *VWorks Automation Control Setup Guide*.

Command	Description
Options	Opens the Options dialog box, which allows you to set global options in the software, including: • View directories and paths. See "Viewing and setting file directory paths" on page 299. • Set general options. See "Setting general and view options" on page 301. • Set email options. See "Setting up email notification" on page 309. • Set view options. See "Setting general and view options" on page 301. • Set error handling options. See "Setting error-handling options" on page 307. • Establish connection to Inventory database. See the *VWorks Automation Control Setup Guide*. • Establish connection to Experiments database. See the *VWorks Automation Control Setup Guide*. • Establish connection to E-Signatures database. See the *VWorks Automation Control Setup Guide*.

Window menu

Command	Description
Cascade	Arranges the protocols and device files in the cascade style.
Tile	Arranges the protocols and device files in the tile style.
Arrange Icons	Aligns the icons in a sequential order.
Close All	Closes all open protocols and device files. If changes were made to the protocols and device files, a message will prompt you to save the changes.
Manage Windows	Opens the Window dialog box and allows you to arrange, minimize, or close the selected protocol and device files.

Help menu

Command	Description
Knowledge Base	Opens the default browser window and displays the Knowledge Base.

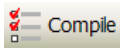
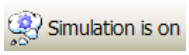
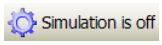
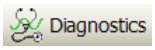
Command	Description
About VWorks	Displays the following: • VWorks software installer number • Copyright dates • List of plugins installed

Toolbar buttons

Standard toolbar

Button	Description
	Creates a new device file or protocol.
	Opens the file you select.
	Saves the changes in the protocol or device file.
	Saves all protocol and device files currently open in the software.
	Removes selected text and stores it in memory.
	Copies selected text and stores it in memory.
	Pastes the text that is currently stored in memory.
	Prints the selected protocol, device, or form file.
	Displays the following: • VWorks software installer number • Copyright dates • List of plugins installed
	Allows to you to display a help topic in context of where you are in the software. Click the Context Help button, and then click a user-interface item, such as a task icon, to display information about that item.

Control toolbar

Button	Description
 or 	Allows you to log in or log out of the software.
	Compiles the selected protocol.
	Starts the protocol run.
	Pauses the protocol run.
 or 	Turns on or turns off the simulation mode.
	Opens the selected device diagnostics plugin.

B Setting up and using the Watcher tool

This appendix assumes that you know how to write programs in JavaScript or have basic programming knowledge. You also must have VWorks administrator or technician privileges.

The topics in this appendixsection are:

- "Watcher overview" on page 756
- "Creating the script that Watcher will run" on page 757
- "Setting up the Watcher configuration file" on page 761
- "Turning on Watcher" on page 765

Watcher overview

CAUTION **Data integrity risk! The Watcher feature may not meet the compliance requirements of your organization. Agilent advises impacted customers not to use the Watcher. The VWorks software does not log an audit trail or provide tamper detection for the Watcher configuration file (.ini). This file type cannot be secured in the Secure Storage Repository.**

Description

Watcher is a tool that automates the processing of designated files using JavaScript. Watcher monitors a designated folder for new file activity, and when a new file appears, Watcher runs a specified JavaScript to process the file. A couple of scenarios are described below, but your script might process files in a variety of other ways.

Scenario 1—Asynchronous post-processing of instrument-generated files

In this scenario, Watcher automates the post-processing of instrument-generated files as follows:

- A protocol includes a reader task that generates an output file and stores the file in a folder that is designated for monitoring by Watcher.
- When the new file appears in the folder, Watcher validates the file and then runs a script.
- The script parses out the relevant information from the instrument-generated file and performs additional tasks, such as aggregating the information across multiple files or saving the information in a database.

Scenario 2—Creating data-driven working protocols from a template protocol

In this scenario, the VWorks software is integrated with a LIMS. Watcher runs a script that automates the creation of a working protocol based on a template protocol and a LIMS-generated input file, where:

- *Input file*. A file that appears in the watched folder. In this scenario, the file specifies attributes of a single protocol run.
- *Template protocol*. A protocol that is used as the basis for creating a working protocol.
- *Working protocol*. The protocol that is created by JavaScript based on the template protocol and the input file.

When an input file appears in the Watcher monitored folder, Watcher runs a script. The script that has to parse the input file must be developed to recognize the file format. The script uses the information from the input file to modify the template protocol into a working protocol, and then schedules the newly created working protocol as part of a runset.

For an example script of this scenario, see "Creating the script that Watcher will run" on page 757.

Workflow for setting up and using Watcher

Step	For this task...	See...
1	Create the JavaScript that Watcher will run.	"Creating the script that Watcher will run" on page 757
2	Set up the Watcher configuration file.	• "Creating the configuration file" on page 761 • "Specifying the configuration file location" on page 763
3	Turn on the Watcher.	"Turning on Watcher" on page 765

Creating the script that Watcher will run

About this topic

This topic assumes that you know how to write programs in JavaScript or have basic programming knowledge.

For a full description of the JavaScript language, see the Mozilla Developer Center at http://www.mozilla.org/js/.

Guidelines for creating your script

In addition to using good script-writing practices, follow these guidelines when creating a script for Watcher to run:

- *Define the JavaScript function correctly*. The function that Watcher will run, for example WatcherMain, must take one parameter:

Parameter	Type	Description
input_file	string	The file path to the input file.

  The name of the function in the script must match the function name in the Watcher configuration file (.ini).

- *Identify or create any other scripts that your script will call*. Agilent has JavaScript files with predefined objects and functions that you may use, including:

Script file name	Description
DeviceArbitration.js	Enables device pooling in JavaScript.

FileUtilities.js	Provides the following functions:
	`isFileExist(filename)`
	`DeleteFile(filename)`
	`StripPath(full_path) //for example, changes "c:/mydir/mysubdir/ myfile.ext" to "myfile.ext"`
	`ForwardToBackSlashes(full_path) //for example, changes "c:/mydir/ mysubdir/myfile.ext" to "c:\mydir\mysubdir\myfile.ext"`

ProtocolEditor.js	Provides various functions for editing a protocol.
StopGo.js	Provides a way to create stop and go tasks using JavaScript, which is useful when stop and go should be scripted based on instance number.
VIN_handling.js	Provides a way to assign virtual instance numbers (VIN) to plates. This is useful if instance numbers: • Must be passed from process to spawned process. • Require an out-of-order assignment, for example, if rejecting some plates, sequential [virtual] instance numbers can be assigned to the plates that remain.
XMLKeyValueLookup.js XMLKeyValueLookup SampleData.xml	Script and sample data for mapping one string to another string, where the definition is in XML format. For example, you might use this script to translate an alias into a final file name.
XML_files.js and Formatter.xml	Provides script for opening XML files, validation against schemas, and saving. Uses MSXMLDOM ActiveX, which requires Msxml2.DOMDocument.6.0 and Msxml2.XMLSchemaCache.6.0.
File_operations.js	Provides functions for various file operations, such as creating, reading, writing to, and debugging files.
plateDB_HowTOUse.js	Provides examples on how to use the VWorks plateDB object, which can be accessed by a script. For a description of the plateDB object, see "plateDB object" on page 140.

- *Define any required global variables, such as folder paths*. You should define the folder paths where processed files will be stored, such as the VWorks folder, the working folder, and an output folder, if applicable. For example, you might define the following:

```
var VWorksFolder    = "C:/VWorks Workspace/"
var WorkingFolder   = VWorksFolder + "WorkingFolder/"
```

```
var OutputFolder    = VWorksFolder + "Output/"
```

- *Include code at the end of the script to delete the input file from the monitored folder, if applicable*. To prevent repeated processing of files upon restarting Watcher, the script should delete the input file or move it into a processed folder or output folder after processing the file.

  Example script:

  ```
  print("deleting input file: [" + input_file_name + "]")
  DeleteFile(input_file_name);
  ```

Script example—Creating data-driven working protocols from a template protocol

This section provides example script for scenario 2 in "Description" on page 756. In this scenario, Watcher monitors a folder for new input files, each of which specifies the attributes of a single protocol run. When a new input file appears in the folder, Watcher runs a script to create a working protocol based on a template protocol and an input file, and then schedules the protocol.

The following script example shows a hypothetical WatcherMain function, which does following:

1. Verifies that the input file exists.
2. Reads the input file.
3. Opens the template protocol.
4. Modifies the template protocol to create a working protocol.
5. Saves the input file and the modified protocol (working protocol) to the working folder.
6. Schedules the working protocol as part of a runset.
7. Deletes the input file from the monitored folder.

To accomplish some of these tasks, the WatcherMain function includes calls to other predefined JavaScript functions, such as inputParser and protocolEditor.

WatcherMain function example

```
function WatcherMain(input_file_name){
   print("Starting WatcherMain...with input file: [" + input_file_name + "]")
   print("OutputFolder : [" +OutputFolder+"]")

   if(!isFileExist(input_file_name))
       Print("input file does not exist");

   var inputParser = new InputParser();
   inputParser.Open(input_file_name);

   var protocol_file_name = inputParser.getOrderAttribute("protocol")
   print("\n protocol file: " + protocolKey);

   print("\n protocol_file_name: " + protocol_file_name);

   var time_string = getTimeString()

   var protocolEditor = new ProtocolEditor();
   protocolEditor.Open(protocol_file_name);

   var working_input_filename = WorkingFolder + time_string + "_" +
    StripPath(input_file_name)
   print("saving working input file to: ["+working_input_filename+"]")
   inputParser.Save(working_input_filename)
   print("success")

   print("modifying protocol")
   ModifyProtocol(inputParser, protocolEditor, working_input_filename);
   print("success")

   var working_protocol_filename= WorkingFolder + time_string + "_" +
    StripPath(protocol_file_name)
   print("saving working protocol to: ["+working_protocol_filename+"]")
   protocolEditor.Save(working_protocol_filename);
   print("success")

   runset.appendProtocolFileToRunset(working_protocol_filename, 1, "this is
    a note: blah", false)


   print("deleting input file: [" + input_file_name + "]")
   DeleteFile(input_file_name);

}
```

Related information

For information about...	See...
Configuring the Watcher feature	"Setting up the Watcher configuration file" on page 761
Creating the JavaScript	"Creating the script that Watcher will run" on page 757
Turning on or off the Watcher feature	"Turning on Watcher" on page 765

Setting up the Watcher configuration file

About this topic

This topic assumes that you know how to write programs in JavaScript or have basic programming knowledge. You must also have VWorks administrator or technician privileges.

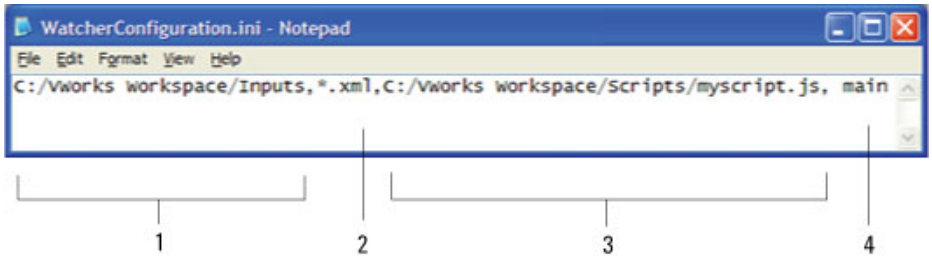
CAUTION **Data integrity risk! The Watcher feature may not meet the compliance requirements of your organization. Agilent advises impacted customers not to use the Watcher. The VWorks software does not log an audit trail or provide tamper detection for the Watcher configuration file (.ini). This file type cannot be secured in the Secure Storage Repository.**

Creating the configuration file

Watcher requires a single configuration file (.ini) that contains one entry (monitoring condition) per line. Each monitoring condition entry must consist of the following four values separated by commas:

<folder to monitor>,<file filter>,<JavaScript file location>, <JavaScript function>

Figure Watcher configuration file showing an example of the four comma-separated values



Item	Value	Description
1	Folder to monitor	The full path to the file folder that Watcher will monitor for new files. Use forward slashes (/) as a path delimiter.
2	File filter	The file name extension of the file type to be monitored. The filter applies to file name extensions only. You cannot filter by partial file names.
3	JavaScript file location	The full path to the JavaScript file. Use forward slashes (/) as a path delimiter.
4	JavaScript function	The function in the JavaScript file that you want to run when the desired files appear in the monitored folder.

Configuration example—Monitoring one folder for one file type

In the following example, Watcher will monitor the Inputs folder for the file type .xml. When an .xml file is added to the folder, Watcher will run myscript.js and call the `main` function. The function `main` must be defined to take in one parameter, the full path of the file that has appeared in the folder to be watched.

- Contents of .ini file:

```
C:/VWorks Workspace/Inputs,.xml,C:/VWorks Workspace/
Scripts/myscript.js, main
```

- Contents of myscript.js file:

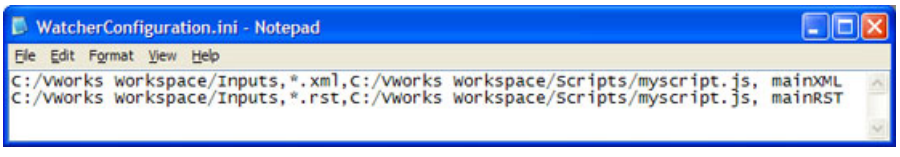
```
function main(input_file){
    print("in main: " + input_file)
}
```

Configuration example—Monitoring one folder for more than one file type

The Watcher configuration file (.ini) can specify multiple monitoring conditions, but each condition must be stated on a single line and consist of the four comma-separated values.

In the following example, Watcher will monitor the Inputs folder for the file types .xml and .rst (runset). When either file is added to the folder, Watcher runs myscript.js and calls the corresponding JavaScript function for the file type. The functions `mainXML` and `mainRST` must be defined to take in one parameter, the full path of the file that has appeared in the folder to be watched.

Figure Watcher configuration file example specifying one folder and two file types for monitoring



For this example, the myscript.js file would contain the following code:

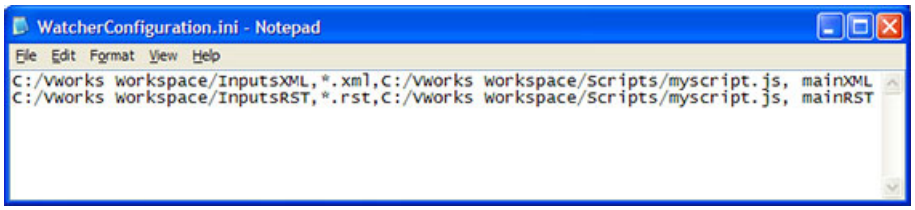
```
function mainXML(input_file){
```

```
    print("in mainXML: " + input_file)
}
function mainRST(input_file){
    print("in mainRST: " + input_file)
}
```

Configuration example—Monitoring more than one folder

You can specify that more than one folder be monitored and run different scripts for different file types that appear in each folder.

Figure Watcher configuration file example specifying two folders and two file types for monitoring



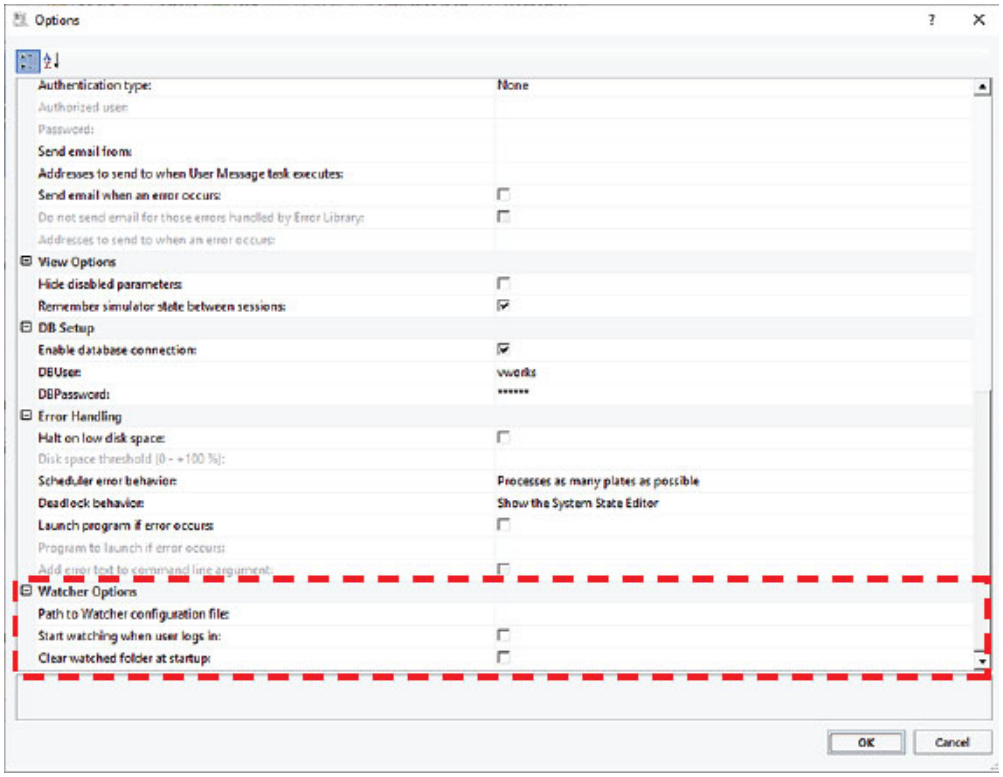
Using the same script from the previous example, this configuration file would do the following:

- Monitor the InputsXML folder for new .xml files, and then call the `mainXML` function. The `mainXML` function must be defined to take in one parameter, the full path of the file that has appeared in the folder to be watched.
- Monitor the InputsRST folder for new .rst files, and then call the `mainRST` function. The `mainRST` function must be defined to take in one parameter, the full path of the file that has appeared in the folder to be watched.

Specifying the configuration file location

To set the configuration file location:

1 Select **Tools > Options.** The Options dialog box opens.



2 In the **Watcher Options** area, click the field next to **Path to Watcher configuration file**, and then click the ⋯ button that appears.

3 In the **Open** dialog box, select the desired location and click **Open.**

4 In the **Options** dialog box, verify the new path, and then click **OK**.

Related information

For information about...	See...
Getting a license for Watcher	Automation Solutions Customer Service
Creating the JavaScript	"Creating the script that Watcher will run" on page 757
Turning on or off the Watcher feature	"Turning on Watcher" on page 765
Reporting problems with the software	"Reporting problems" on page 743

Turning on Watcher

About this topic

This topic assumes that you have VWorks administrator or technician privileges.

CAUTION **Data integrity risk! The Watcher feature may not meet the compliance requirements of your organization. Agilent advises impacted customers not to use the Watcher. The VWorks software does not log an audit trail or provide tamper detection for the Watcher configuration file (.ini). This file type cannot be secured in the Secure Storage Repository.**

About monitoring new files

The first time you turn on Watcher, the program runs the specified JavaScript for the existing files in the folder that is configured for monitoring. Subsequently, Watcher processes only the new files that appear in the folder.

Watcher will not process existing files whose contents were modified. For example, if a new file replaces an existing file with the same file name, Watcher does not reprocess the file. If you rename an existing file, Watcher processes the renamed file as a new file.

Turning on and turning off Watcher

You can turn on Watcher manually or automatically every time you log in to the VWorks software. You can turn off Watcher manually.

To turn on Watcher manually:

1 Select **Tools > Watcher is OFF**. The menu command changes to Watcher is On.

To turn on Watcher automatically every time you log in:

1 Select **Tools > Options.** The Options dialog box opens.

2 In the **Watcher Options** area, select **Start watching when user logs in.**

The next time a user logs into the VWorks software, Watcher will turn on automatically and remain on after the user logs out.

To turn off Watcher:

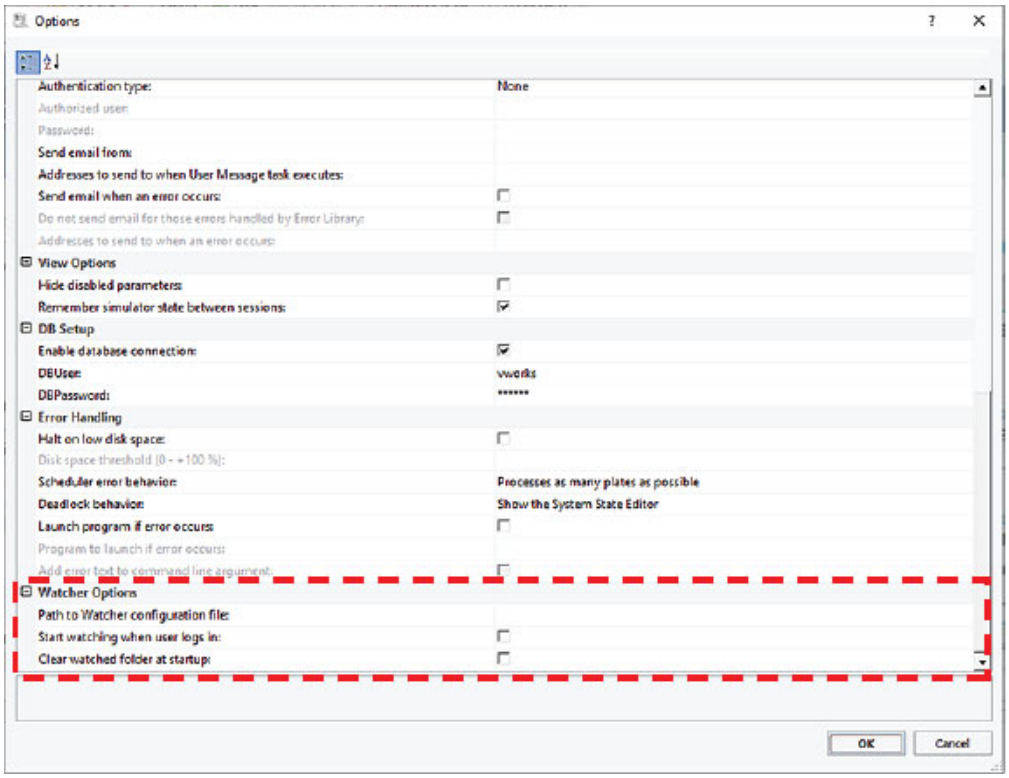
1 Select **Tools > Watcher is On**. The menu command changes to Watcher is Off.

To prevent Watcher from turning on automatically when a user logs in:

1 Select **Tools > Options.** The Options dialog box opens.

2 In the **Watcher Options** area, clear the **Start watching when user logs in** check box.

Figure Options dialog box showing the Watcher Options area



Related information

For information about...	See...
Reporting problems	“Reporting problems” on page 743
Log file directory settings	“Viewing logs” on page 714
General and view options	“Setting general and view options” on page 301
Error-handling options	“Setting error-handling options” on page 307
Starting a protocol run	“Starting the protocol run” on page 312

In This Book

This guide explains how to use the VWorks Automation Control software to

- Set up devices
- Write protocols
- Run protocols
- Troubleshoot protocols
- Operate devices using VWorks software

www.agilent.com

© Agilent Technologies, Inc. 2024

July 2024

D0004378 Revision B.2

Agilent